ACOUSTIC MODELLING, DATA AUGMENTATION AND FEATURE EXTRACTION FOR IN-PIPE MACHINE LEARNING APPLICATIONS

D A CHIANTELLO

PhD 2022

ACOUSTIC MODELLING, DATA AUGMENTATION AND FEATURE EXTRACTION FOR IN-PIPE MACHINE LEARNING APPLICATIONS

DARIO ALFREDO CHIANTELLO

A thesis submitted in partial fulfilment
of the requirements of
Manchester Metropolitan University
for the degree of Doctor of Philosophy

Department of Engineering
Manchester Metropolitan University

2022

Declaration

I, Dario Alfredo Chiantello, born in Catania (Italy) on 13/02/1982, declare that this thesis and the work herein presented in it are my own and the result of my original research investigations. I hereby state that this work has not been submitted, in whole or in part, in any previous application for a degree and is not currently being submitted in candidature for any degree other than the one of Doctor of Philosophy (PhD) at Manchester Metropolitan University (MMU).

Dario Alfredo Chiantello

Acknowledgements

At the end of this long journey, my first thought is certainly for my family and, in particular, for my parents. A considerable amount of time dedicated to my PhD has also been time spent away from them, and this is probably the best way to quantify the extent of my sacrifice. I'm sure the frustration I sometimes experienced has partially spilt over, causing sorrow and distress in their lives. The same thought is also for Masha and all my other close friends, who, to some extent, have felt the impact of a choice that I erroneously believed could be strictly personal. As much as I could, I tried not to neglect all those who largely define who I am, but part of the time I spent on my research was inevitably stolen from them.

I started this adventure with the desire to invest my time for a better future and also because I was inspired by the personal experience of a friend and former colleague of mine, Dr Guillermo Jimenez. If today I'm writing these lines, it is also because of him. I'm not sure if I should hate or love him for that, but I'm glad we remained in contact during these years, sharing and discussing our difficulties along our respective journeys.

I first expressed my desire to submit a research proposal during my interview for a KTP position at Manchester Metropolitan University. On that occasion, I met my supervisor, Professor Bamidele Adebisi. For him, I reserve a token of gratitude for the trust and the support I received, initially as a KTP associate and later as a PhD student. Together we managed to complete what we aimed to achieve in our original plan, also sharing the great satisfaction for the Best KTP Award 2020 and second place for the Best Engineering in the same contest. I'm conscious that my attitude and my working style are sometimes difficult to deal with. Nevertheless, during these years, I have always found his door open, receiving precious advice without feeling obliged to fight my personal nature. I believe this was not always straightforward for him. I have certainly noticed and appreciated his effort to cope with my deficiencies.

On top of my acknowledgements, there is also Aquacheck Engineering Ltd and the whole fantastic team. Without them and their financial support, my personal

background would have certainly been poorer today. Special thanks go to the CEO Paul Carrington, for leaving me the freedom to organise my time unavoidably divided between my PhD and the office. Sharing and discussing new ideas together has been an enjoyable exercise throughout my years with them. I tried to compensate for the support I received by offering my full commitment and fairness, first as a KTP associate and later as an employee. Nevertheless, I'm certainly aware that my role and my outcomes for the company have been profoundly affected by the time and mental commitment required for my research. A special mention goes to Dr Trevor Gregory and also to Dr David Woollard, who sadly passed away in 2020. Their support was crucial during my KTP and certainly also relevant for my PhD.

My acknowledgements cannot miss those who kindly agreed to spend a few hours of their time providing their valuable feedback on my work. Apart from my supervisor, I would like to thank Dr Kyungmin Baik for his useful observations on the acoustic problems of my research, and Dr Segun Popoola for the interesting conversations on machine learning. My appreciation also to Dr Weizhuo Wang, Dr Sunday Ekpo, and Dr Muhammad Ijaz for the beneficial advice received during the annual reviews. Along with them, my regards to the board of examiners of my viva, Professor Lyudmila Mihaylova, Dr Rasool Erfani, and Dr Sanja Potgieter, for their respectful and professional approach.

My final remark is for myself and for other students who might read this work. Achieving my PhD required commitment and strong motivation, and, despite all the limitations, I'm certainly proud of this result. Nevertheless, I believe that most of the frustrations I experienced were related to a research proposal that required some adjustment halfway through and, more importantly, to the limited number of occasions I could, given my specific condition, discuss and share the results. I'm fairly certain that a deeper assessment of these issues can make the overall PhD journey more enjoyable and profitable.

Contents

ACOUSTIC MODELLING, DATA AUGMENTATION AND FEATURE EXTRACTION FOR IN-PIPE MACHINE LEARNING APPLICATIONS

Dario Alfredo Chiantello

Abstract

Gathering measurements from infrastructure, private premises, and harsh environments can be difficult and expensive. From this perspective, the development of new machine learning algorithms is strongly affected by the availability of training and test data. We focus on audio archives for in-pipe events. Although several examples of pipe-related applications can be found in the literature, datasets of audio/vibration recordings are much scarcer, and the only references found relate to leakage detection and characterisation. Therefore, this work proposes a methodology to relieve the burden of data collection for acoustic events in deployed pipes. The aim is to maximise the yield of small sets of real recordings and demonstrate how to extract effective features for machine learning. The methodology developed requires the preliminary creation of a soundbank of audio samples gathered with simple weak annotations. For practical reasons, the case study is given by a range of appliances, fittings, and fixtures connected to pipes in domestic environments. The source recordings are low-reverberated audio signals enhanced through a bespoke spectral filter and containing the desired audio fingerprints. The soundbank is then processed to create an arbitrary number of synthetic augmented observations. The data augmentation improves the quality and the quantity of the metadata and automatically creates strong and accurate annotations that are both machine and human-readable. Besides, the implemented processing chain allows precise control of properties such as signal-to-noise ratio, duration of the events, and the number of overlapping events. The inter-class variability is expanded by recombining source audio blocks and adding simulated artificial reverberation obtained through an acoustic model developed for the purpose. Finally, the dataset is synthesised to guarantee separability and balance. A few signal representations are optimised to maximise the classification performance, and the results are reported as a benchmark for future developments. The contribution to the existing knowledge concerns several aspects of the processing chain

implemented. A novel quasi-analytic acoustic model is introduced to simulate in-pipe reverberations, adopting a three-layer architecture particularly convenient for batch processing. The first layer includes two algorithms: one for the numerical calculation of the axial wavenumbers and one for the separation of the modes. The latter, in particular, provides a workaround for a problem not explicitly treated in the literature and related to the modal non-orthogonality given by the solid-liquid interface in the analysed domain. A set of results for different waveguides is reported to compare the dispersive behaviour against different mechanical configurations. Two more novel solutions are also included in the second layer of the model and concern the integration of the acoustic sources. Specifically, the amplitudes of the non-orthogonal modal potentials are obtained using either a distance minimisation objective function or by solving an analytical decoupling problem. In both cases, results show that sources sufficiently smooth can be approximated with a limited number of modes keeping the error below 1%. The last layer proposes a bespoke approach for the integration of the acoustic model into the synthesiser as a reverberation simulator. Additional elements of novelty relate to the other blocks of the audio synthesiser. The statistical spectral filter, for instance, is a batch-processing solution for the attenuation of the background noise of the source recordings. The signal-to-noise ratio analysis for both moderate and high noise levels indicates a clear improvement of several decibels against the closest filter example in the literature. The recombination of the audio blocks and the system of fully tracked annotations are also novel extensions of similar approaches recently adopted in other contexts. Moreover, a bespoke synthesis strategy is proposed to guarantee separable and balanced datasets. The last contribution concerns the extraction of convenient sets of audio features. Elements of novelty are introduced for the optimisation of the filter banks of the mel-frequency cepstral coefficients and the scattering wavelet transform. In particular, compared to the respective standard definitions, the average F-score performance of the optimised features is roughly 6% higher in the first case and 2.5% higher for the latter. Finally, the soundbank, the synthetic dataset, and the fundamental blocks of the software library developed are publicly available for further research.

Nomenclature

General notations

$\mathbb{N}$	Natural numbers starting from 1
$\mathbb{Z}$	Integer numbers
$\mathbb{R}^{N \times M}$	Real numbers in $N \times M$ dimensions
$\mathbb{C}^{N \times M}$	Complex numbers in $N \times M$ dimensions
$\forall$	For all
$\in$	In
$\mid$	Such that
i	Unit imaginary number
a^*	Complex conjugate of a
$\Re(a)$	Real part of complex number a
$\Im(a)$	Imaginary part of complex number a
$g(t)$	Function of continuous variable t
$g[n]$	Function of discrete variable n
$\tilde{g}$	Approximation of g
:	Full range
$n : m$	Integer indexes in the range between n and m
$a : w : b$	Values between a and b in steps of w
$\boldsymbol{x}$	Vector or matrix of elements
$\boldsymbol{x}_{(A,B)}$	Elements of matrix $\boldsymbol{x}$ in row range A and column range B
$\boldsymbol{x}^T \boldsymbol{y}$	Inner product of column vectors $\boldsymbol{x}$ and $\boldsymbol{y}$
$d(\boldsymbol{x}, \boldsymbol{y})$	Distance between vectors $\boldsymbol{x}$ and $\boldsymbol{y}$
$\|\boldsymbol{x}\|_p$	p-norm of $\boldsymbol{x}$
$\|\boldsymbol{x}\|_F^2$	Frobenius norm of $\boldsymbol{x}$
$\boldsymbol{x}^T$	Transpose of $\boldsymbol{x}$

$\boldsymbol{x}^{\dagger}$	Moore-Penrose pseudoinverse of $\boldsymbol{x}$
$\mathscr{F}$	Fourier transform operator
$\hat{g}(\omega)$	Fourier transform of function $g(t)$
$\lvert\hat{g}(t,f)\rvert$	Spectrogram of $g(t)$
$g \star h$	Convolution of functions g and h
$corr(g,h)$	Correlation of functions g and h
$\langle g,h \rangle$	Inner product of functions g and h
δ	Dirac delta
δ_{nm}	Kronecker delta for indexes n and m

Acoustic notations

$\vec{\boldsymbol{u}}$	Unitary field versor
$\vec{\boldsymbol{v}}$	Field vector
∇^2	Laplace operator
∇a	Gradient of scalar field a
$\nabla \cdot \vec{\boldsymbol{v}}$	Divergence of field vector $\vec{\boldsymbol{v}}$
$\nabla \times \vec{\boldsymbol{v}}$	Curl of field vector $\vec{\boldsymbol{v}}$
$\vec{\boldsymbol{v}} \cdot \vec{\boldsymbol{w}}$	Dot product of field vectors $\vec{\boldsymbol{v}}$ and $\vec{\boldsymbol{w}}$
J_n	Bessel function of the first kind of order n
Y_n	Bessel function of the second kind of order n
$\Delta^2(g,h)$	Distance between functions g and h
p_0	Dimensionless pressure
$\hat{p}_x$	Pressure transfer function

Machine learning notations

$\{\}$	Homogeneous group of elements
$\mathbb{E}$	Expected value operator
μ, η	Mean value
σ	Standard deviation
σ^2	Variance
s_n	Statistical moment of order n

$\mathscr{D}$	Dictionary of atoms
ζ_γ	Time-frequency atom waveform
ϕ	Windowing function
Cx	Autocorrelation of signal $x(t)$
$Z_{cr}x$	Zero-crossing rate of signal $x(t)$
ψ	Mother wavelet
$\psi_{u,s}$	Child wavelet with shift u and dilation s
Wx	Wavelet transform operator for signal x
$\|W_m\|x$	Wavelet modulus transform operator for layer m and signal x
$S_m x$	Wavelet scattering operator for layer m and signal x
Q	Filter quality factor
$\mathscr{L}$	Lagrangian function
$\boldsymbol{\xi}$	Map function
κ	Kernel function
$\boldsymbol{K}$	Gram matrix
$l_b(\boldsymbol{x}, j, k)$	Binary loss for example $\boldsymbol{x}$, classifier j, and class k
$l(\boldsymbol{x}, k)$	Negative loss for example $\boldsymbol{x}$ and class k
$m(\boldsymbol{x})$	Classification margin for the example $\boldsymbol{x}$
E	Classification edge
O	Computational complexity order of approximation

Acronyms and abbreviations

FE	Finite elements
FD	Finite differences
FFT	Fast Fourier transform
$IFFT$	Inverse fast Fourier transform
PSD	Power spectral density
$STFT$	Short-time Fourier transform
$MFCC$	Mel-frequency cepstral coefficients
WST	Wavelet scattering transform
PCA	Principal component analysis

PC	Principal component
KNN	K-nearest neighbors
SVM	Support vector machine
TP	True positive
FP	False positive
TN	True negative
FN	False negative
Acc	Accuracy
Err	Error rate
Fsc	F-score

Physical quantities

λ	Wavelength
f	Frequency
ω	Angular frequency
c	Sound velocity
ρ	Density
$\boldsymbol{\sigma}$	Stress tensor
σ_{hh}	Normal stress component in the h direction
σ_{hj}	Shear stress j in respect of the normal direction h
I_{σ_h}	h^{th} stress invariant
$\boldsymbol{\varepsilon}$	Strain tensor
ε_{hh}	Normal strain component in the h direction
ε_{hj}	Shear strain j in respect of the normal direction h
I_{ε_h}	h^{th} strain invariant
$\vec{\boldsymbol{b}}$	Body forces
$\vec{\boldsymbol{t}}^{(x)}$	Traction vector in the x direction
$\vec{\boldsymbol{u}}$	Particle displacement vector
$\vec{\boldsymbol{v}}$	Particle velocity vector
E	Young's modulus
ν	Poisson's ratio
μ	Shear modulus
K	Bulk modulus
λ, μ	Lamé constants
$\vec{\boldsymbol{k}}$	Wavenumber vector
k_z	Axial wavenumber component
$k_{z_{nm}}$	Axial wavenumber component for mode (n,m)

q	Orthogonal wavenumber component
q_{nm}	Orthogonal wavenumber component for mode (n,m)
ϕ	Displacement scalar potential for the inner liquid
Φ	Frequency domain amplitude of the potential ϕ
γ	Displacement scalar potential for the solid shield
Γ_j	j^{th} frequency domain amplitude for the potential γ
$\vec{\boldsymbol{\psi}}$	Displacement vector potential for the solid shield
Ψ_j	j^{th} frequency domain amplitude for the potential $\vec{\boldsymbol{\psi}}$
p	Pressure
v_p	Phase velocity
v_g	Group velocity
Q	Source strength
z	Specific acoustic impedance

List of Figures

List of Tables

Chapter 1

Introduction

This chapter briefly overviews the thesis background, motivations, and objectives. The most significant contributions are listed to point out the elements of novelty developed. The last section briefly summarises the content of each chapter, the connections between them and their role in respect of the objectives stated.

1.1 Background and motivations

Water infrastructures play a key role in modern society, and in a scenario of growing water constraints, the quest for efficient management is more crucial every day. This section briefly analyses some of the fundamental causes of water scarcity, how they relate to our society, and the importance of the infrastructures for mitigating future challenges.

1.1.1 Emergent constraints on water availability

In recent years, the need to account for climate change when managing natural resources has become more relevant, and water is no exception [1]. The pattern of precipitations has undoubtedly mutated, and there is a fear of more significant alterations in the foreseeable future [2]. Although different regions worldwide have been affected differently, changes in precipitation intensity, duration, and frequency have often translated into reduced snow coverage [3],[4] and shorter and more

violent rainfalls [5]. This shift has determined a soaring number of floods and increased the pressure on water reservoirs, better preserved by regular, cold snow seasons and prolonged, mild rainfalls [6]. Extreme weather conditions in arid macro-areas, such as Mediterranean countries, are being exacerbated by drier summers, more frequent flooding and an overall higher number of dry days [7]. At the same time, in regions traditionally immune, such as northern Europe, an increased number of torrid periods have already caused unprecedented water usage restrictions and other unusual environmental issues [8].

Apart from climate change, water resources are also affected directly by other human activities. Agriculture and livestock farming are certainly the most relevant, accounting for 85% of the total freshwater requirements [9, p. 15],[10]. They are also frequently related to other serious issues, such as soil salinisation [11], increased levels of pollutants in underground aquifers [12], and the depletion of non-renewable fossil water [13]. Moreover, the demand from other industrial sectors, such as renewable energy production, is growing fast [14].

Amid this extensive need in the context of scarcer availability, growing populations in large urban clusters require access to vast natural resources in a limited space [15]. In dry regions, there is already a tangible fear that a lack of adequate access to fresh water might soon lead to political instability or even conflicts [11],[16]. Where growing cities have failed to introduce new effective water management policies, the consequences on the existing water reservoirs are concerning, going from worsening water quality to pronounced level drops in the underground aquifers [17]. Alternative solutions, such as desalination plants, have been extensively adopted in countries affected by chronic water scarcity. Still, management costs and controversial environmental issues introduce geographical limitations and limit their applicability to a fraction of the required water [18].

Given the complex nature of the problem and the number of issues involved, a strategy to guarantee efficient usage of the resources appears necessary. Such an approach, however, requires information that cannot be gathered without enhanced infrastructures.

1.1.2 Water scarcity mitigation through enhanced infrastructures

The introduction of effective water management policies is fundamental to mitigating the impact of water scarcity; several examples of virtuous big cities are reported in the literature [19].

Although it has been shown that soft strategies, such as tariff modulation or raising user awareness, play a key role in inducing sensible behaviours [20], adopting new technologies is fundamental to support the process. The solutions employed span from simple plumbing improvements, such as more efficient shower heads and low-flush toilets [20], to more sophisticated technologies, such as leakage detectors [21] and satellite imagery [22]. For instance, installing smart water meters at the household level provides monitoring capabilities with time and space resolution that enable better characterisation and modelling of the water demand [23].

In general, the convenience of a data-driven water management approach has been recognised from different perspectives, from enhanced services to better management policies [24],[25]. Improved maintenance scheduling, lower repairing costs, and reduced unaccounted water consumption are a few examples. Data for such a variety of tasks, however, needs to be detailed and heterogeneous. Unfortunately, although monitoring is generally continuously performed at the source and the main nodes of the network, the status of the peripheral regions is often unknown or irregularly acquired [26]. Monitoring the terminal branches is usually a manual process that requires a large amount of human labour. For instance, water quality is continuously checked at the source but only periodically elsewhere. Hundreds of thousands of water samples per year [27],[28] need to be manually collected, gathered in chemical labs and analysed [29, p. 188]. The absence of automatic acquisition of the network status determines the lack of effective means for the proactive identification of potential danger and less efficient maintenance procedures [30]. Inconveniences, such as leakages, chemical pollutants, and discolouration, are likely related to detectable events along the pipelines, but often suppliers are notified by their customers. [31].

Unfortunately, enhancing the network through the deployment of sensors is generally difficult and expensive. The scale of the required operations is undoubtedly a first barrier but not the only one. Indeed, water is an essential resource, and the risk of potential disruption of the distribution service must be carefully accounted for and mitigated. Furthermore, weak customer support, lack of expertise, and slow policy advancements are among the main reasons for the slow adoption of new solutions and technologies [32]. Because of these difficulties, even testing new devices or collecting sample data from the field might be problematic, especially when the intervention to install the sensors is invasive or requires a service interruption.

A comprehensive assessment of costs and benefits related to new monitoring technologies must also account for non-technological constraints. Privacy concerns, for example, have been investigated extensively for energy metering [33],[34] and also need to be accounted for water [32]. Additional issues, such as accessing private premises, should be considered. Generally, gathering data from fewer sensors placed at accessible points might be beneficial.

Enhancing the water networks is, therefore, a process whose trade-off is driven by the necessity to optimise the usage of an ever-increasingly valuable resource and the difficulties related to the development and deployment of the required new technologies. This work accounts for some of these barriers: it focuses on a specific class of sensors for which the in-house development based on small collections of on-field data samples is the only viable solution. It will be shown how new devices can be developed by optimising the yield of minimum-size datasets.

1.2 Thesis aim and objectives

This work has been inspired by the need to detect hazardous manoeuvres on the water network associated with the usage of standpipes [35]. Monitoring standpipe sessions, for instance, by acquiring their position, time and other details, is crucial to preventing and identifying potential costly damages, reducing water wastage, and obtaining accurate billing [36]. Nevertheless, events in water infrastructures

can stem from a variety of different sources, and automatic detectors/classifiers are fundamental blocks of the smart networks described in the previous section[1]. Therefore, this work generalises the original idea by investigating the process of developing such devices for in-pipe applications.

Among all the possible strategies to detect/classify specific classes of events, we focus on characteristic *acoustic fingerprints* as the means to discriminate among different classes. Unfortunately, on-filed development is generally prohibitive, and the in-house approach often remains the only possible option. Besides, acquiring the audio samples required is typically a costly and time-consuming process, and it is necessary to optimise the yield of small datasets [38, p. 159]. Finally, building test rigs might be significantly expensive, and solutions to enable real tests at a low cost are highly desirable.

***

This work focuses on machine learning applications for in-pipe acoustic events. The main research aim is the definition of a methodology that makes the development of such applications possible when the amount of data available is limited.

***

The achievement of the main research aim can be formulated as five complementary research objectives:

O_1 The first objective is to define the research direction through a comprehensive literature review in acoustics and machine learning, providing the theoretical foundations required.

O_2 The second objective is to provide a methodology that simplifies the acquisition of the real pipe audio samples required, and to create an example of soundbank suitable for generating synthetic audio observations enhanced with accurate annotations and containing specific time-localised events.

[1] Although the published literature references are poor, the need to monitor standpipe sessions is well understood among water service providers. This project, for instance, has been inspired by the development of an IoT device proposed by the funder for the purpose [37]. See also chapter 7.

O_3 The third objective is to provide a methodology to turn the recorded weakly-labelled pipe audio files into datasets of augmented strongly-labelled synthetic observations where the usage of the real source data is optimised, and the properties of the events can be tuned as necessary.

O_4 The fourth objective is to provide a software tool to simulate the reverberation experienced in in-pipe propagation and to use such a tool for audio synthesis. The audio model should also match simple, low-cost, in-house, real test rigs.

O_5 The fifth objective is to define an optimised set of features (or the procedure to extract them) capable of representing the acoustic events for the purpose of feeding machine learning algorithms for event classification and detection. A performance benchmark is also required for future developments on the synthesised dataset.

By achieving the objectives above, it is possible to define a workflow where, from a relatively small number of real data samples, it is possible to develop a working device away from the actual infrastructures and reduce the tuning required after deployment. Then, once the first set of working devices has been deployed, the same workflow can be repeated using more accurate data for a further in-house improvement loop.

Given the commercial constraints of the private funder and the need to provide open access to the data, this work refers to a set of audio samples recorded in a domestic environment for the purpose. The methodology adopted, however, can be generalised to different classes of in-pipe audio events once a suitable sound bank is acquired.

1.3 Literature gaps and major contributions

The contribution of the present research to the existing knowledge can be analysed from different points of view and in relation to the objectives stated above. The related data and the main blocks of the software library created are available to download[2].

[2] Soundbank and dataset: https://doi.org/10.5281/zenodo.7615371
Software library: https://github.com/pipesoundlibrary

O_1 - The literature on machine learning abounds with data archive examples for specific applications [39],[40]. Indeed, different machine learning algorithms share the need for training data as an indispensable prerequisite. Preparing appropriate datasets, however, is a task that can be highly time-consuming and error-prone [38, p. 154],[41]. In addition, acquiring and sorting data beyond common daily experience generally requires special equipment and trained human personnel. Services like *Amazon Mechanical Turk* [42] or *Cloudfactory* [43] aim to reduce the development cost, but the result might be inaccurate and dependent on the subjective experience of the operators [44]. For instance, to manually generate the simplest form of strong annotations for unclassified audio files, it is necessary to listen to the recordings, find the beginning and the end of the audio events, and assign descriptive labels. This operation requires a time longer than the cumulative duration of all the recordings, even when labelling is executed directly at *the point of collection*. Besides, even basic metadata, such as start-time markers, end-time markers, and class labels, are generally affected by human skills and attitudes [45]. To mitigate the issues above, the approach used in this work is articulated in two phases, which relate to the objectives stated in the previous section: the *creation of the soundbank* and the actual *dataset synthesis*.

O_1, O_2 - Although source data are often not directly accessible, several references to datasets for pipe monitoring can be found in the literature and regard various frameworks, such as *ultrasonic-guided monitoring* [46] and *flow and pressure data analysis* [47]. However, archives of pipe audio/vibration recordings are much scarcer, and the only references found relate to leakage analysis [48],[49],[50]. Indeed, within the goal of collecting sample signals, the work we believe is the closest to the scope of the present research deals with audio data for leakage characterisation (*ROPP* dataset) [51]. With regard to the latter, a few important issues must be pointed out to understand the knowledge gaps and the reasons behind objective **O_2** stated in the previous section. Apart from being specific to leakage, in the perspective of the present work, the main issue with the ROPP dataset is its unsuitability for synthesising dynamic and precisely time-localised audio events. Indeed, for this purpose, the recorded audio should include the non-

stationary part of the signals (i.e. on-set and off-set) and the means to separate the background from the relevant audio foreground. The goal is to obtain a dataset that should be suitable not only for classification but also for detecting single and multiple overlapping events. Besides, the recording approach adopted in this work gathers the signals directly from the pipe, a condition closer to the one that can be realised in the final deployable sensors. For what concerns noise filtering, that is, the process of separating the meaningful audio foreground from the background noise, an improved algorithm is proposed with respect to the closest and most recent example found in the literature [52]. Noise filtering, however, can be declined in many flavours [53]. As explained in chapter 5, the algorithm proposed is meant to be compared only against other spectral filters, which we believe is the class of filters most suited for the scope of this research. It is further remarked that the solution proposed for creating the soundbank requires only recording the desired sounds and assigning weak labels, that is, saving audio files belonging to the same class/instance under the same folder. No further relevant manual operation is required. This solution simplifies the acquisition and limits human errors without affecting the quality of the final synthesised dataset.

O_1, O_3 - Objective **O_3** concerns the synthesis of multiple audio observations organised in datasets. In machine learning, artificial audio synthesis is a well-established technique used in several fields of audio processing, such as sound event detection (*SED*) for environmental monitoring [54] and automatic speech recognition (*ASR*) [55]. Frequently, the reason behind the adoption of this approach is the development of machine learning models that are more robust under different conditions, for instance, variable signal-to-noise ratio [56] and reverberation [57]. Examples of artificial synthesis for pipe-related sound scenes are also reported in the literature, being the most relevant work found related to the analysis of leakage sound scenes under different conditions of noise and disturbances [51]. Regarding the software tools, the open-source library *Scaper* [58] seems to be the example of a synthesiser closest to the scope of this work. Scaper allows controlling audio properties, such as signal-to-noise ratio, duration of the events, and the number of overlapping events, all assigned according to probabilistic rules in predefined

ranges. Moreover, the variability of the source events is extended by using a time-invariant pitch shift and frequency-invariant time dilation. However, despite several similarities, the solution here proposed for synthesising in-pipe audio scenes differs under a few crucial aspects related to both the general approach for the synthesis and the specific in-pipe scenario. For instance, with regard to the general approach for the synthesis, the proposed software creates new events by combining chunks of different source observations. This solution provides higher variability on top of time-dilation and frequency shift without introducing any distortion. Checks are performed to promote natural-like results. As for Scaper, labels are organised using the *.jams* format, which allows storing complex metadata while maintaining human and machine readability. However, in the proposed implementation, references are strictly maintained to provide comprehensive traceability of the synthesis and retrieve the corresponding source recordings in the soundbank. Finally, synthetic observations are generated using the *single instance foldable synthesis*, a procedure here introduced to guarantee balanced partitioning of the final dataset. Further details are reported in chapters 3, 5 and 6.

O_1, O_4 - Another main difference and a feature that accounts for the specific in-pipe scenario is certainly the integration of simulated reverberations, as stated by objective **O_4**. Simulating reverberations finds many applications in the literature, from video game audio rendering [59],[60] to understanding human sound perception in closed environments [61]. In machine learning, common examples of applications are the enhancement of indoor speech recognition [62],[63] and the improvement of underwater acoustic detection performances [64]. In-pipe propagation is also affected by reverberation phenomena induced by the presence of geometric boundaries and different materials [65, p. 246]. Assessing the related distortions is a fundamental problem in applications such as in-pipe data transmission [66],[67] and non-destructive health assessment of infrastructures [68]. In general, synthetic reverberations are usually obtained using three different approaches: calculating the convolution between "dry" input signals and a set of recorded impulse responses [38, pp. 161], using delay networks, and using fully numeric/analytic physical models [69]. These techniques are sometimes used in

combination. Despite potentially yielding more realistic results, recording impulse responses can be a tedious task [67]. Indeed, a separate acquisition is required per each configuration of the audio scene [70]. Therefore, accounting for the limitations, alternative solutions based on fully analytic/numeric approaches can sometimes be more effective. For instance, a numeric finite-difference time-domain wave solver is adopted in [71] to calculate the required impulse responses, removing the need for real environment-dependent audio acquisition. In this work, a simulated approach based on a physical model has also been adopted for calculating the impulse responses since, despite the limitations, significant results can be obtained without additional hardware costs or human labour [69]. Generally, this solution is preferred when access to the actual sound scene is difficult or when alterations of the analysed set-up should be allowed (e.g. changing the relative position between source and receiver). Although a perfect simulation of the real environment is unfeasible even with fully numerical methods, a simplified representation might be enough to account for the propagative distortions. Besides, a simple set-up can be replicated in a simple test rig to verify the results. Modelling in-pipe reverberation has long-tracked records [72], being the modal analysis and the characterisation of the boundary conditions two central issues [73, p. 464]. The analyses are generally developed under the theory of linear elasticity [74], sometimes in the *thin-shell* approximation [75]. Application of this framework for creating a simulation tool for datasets of in-pipe audio events has, to the best of our knowledge, no previous example in the literature. Therefore, we derive a novel model based on the linear elasticity theory that, although less flexible than finite element solutions, is much faster to calculate, easily integrable into the synthesiser, and ready to be matched to a cheap test rig. With respect to other solutions using a similar approach [67],[76], a novel algorithm is proposed to automatically extract, sort and interpolate the roots of the characteristic elasticity equation. Moreover, two novel solutions are proposed for calculating the modal amplitudes under the non-orthogonality condition dictated by the solid-liquid boundary of the waveguide. Finally, the main factors affecting the reverberation are analysed along with a set of rules to perform the measurements in a real test rig. Further details are reported in chapters 2 and 4, while the integration of the model for the synthesis of the

dataset is reported in chapter 5.

O_1, O_5 - The release of new datasets, including those for SED, is generally combined with the release of benchmarks used as references to assist the development of new machine learning algorithms and assess the related performance [77],[78]. Therefore, as stated by objective **O_5**, we also provide a set of reference results for classification obtained using a processing chain with multiple configurations. The development of new classification algorithms often focuses either on extracting a better set of features or improving the classifier solution. However, in the latter case, the choice of a specific set of features is rarely justified [38, p. 96]. Although mainly dictated by the context of the research, this approach is potentially sub-optimal since extracting features generally causes the loss of a certain amount of information and the possible introduction of biased representations [79]. Therefore, this work primarily focuses on obtaining an optimised set of features and leaves a deeper analysis of the classifier problem for further research. We reviewed the literature to select feature representations that provide better performance in terms of loss of information, stability, invariance, and computational requirements; then, we optimised such features in respect of the dataset developed. In particular, optimised representations are proposed starting from *short-time Fourier transform (STFT)* [80, pp. 101–102], *mel frequency cepstral coefficients (MFCC)* [81, pp. 87–91], and *wavelet scattering transform (WST)* [82]. The STFT offers a baseline reference with a fundamental signal representation. MFCC provide better performance while maintaining low computational complexity. To improve the structure of the filter bank, generally optimised to emulated bio-inspired characteristics [83], the role of the windowing function is investigated and a novel algorithm (inspired to *pattern search* [84] and *simulated annealing* [85]) is proposed to optimise the number of filters and the mel-Hz conversion law. The WST is adopted because of its capability to retain the information of the source signal at the price of higher computational complexity [79]. An optimisation is proposed for the number of filters and for the selection of the filters' frequency boundaries in relation to the Nyquist frequency. *K-nearest neighbours (KNN)* and *support vector machine (SVM)* classifiers are used for classification. The former for its simplicity [38, p. 112] while the latter for

the good performance with low computational complexity [86] and for the possibility to exploit the separation margin to optimise the feature representation [87]. Results are provided for single and multi-folds partitions under a non-overlapping hypothesis. Further details are reported in chapter 6.

1.4 Thesis outline

The structure of this work (figure 1.1) is dictated directly by the objectives reported in section 1.2. It articulates in four main blocks: *the definition of the acoustic model* (chapters 2 and 4), *the creation of the soundbank* (chapters 3 and 5), *the synthesis of the datasets* (chapters 3, 5, and 6), and *the machine learning features and benchmark* (chapters 3 and 6). A final conclusion chapter is further reported at the end. Chapters 2 and 3 provide the necessary theoretical background and the technical details behind the motivations for the novel research developed in chapters 4, 5 and 6.

More in detail, *chapter 2* introduces the basic concepts of the linear elasticity theory together with the related wave equations and the analysis of the relevant boundary conditions. The main physical quantities generally used to describe dispersion phenomena are also defined. Finally, the equations for acoustic propagation inside a waveguide are derived for the rigid and the elastic boundary case.

Chapter 3 provides fundamental notions of machine learning and introduces the concepts required to build a minimal but complete processing chain: from datasets to performance indicators. The motivation behind the adoption of certain signal representations and the related transformations are given together with the strengths and the weaknesses of the classification techniques used.

Chapter 4 introduces the novel acoustic model to simulate the acoustic propagation in a pipe. It provides a description of the algorithm created to extract and sort the roots of the characteristic elasticity equation and the methodologies to determine the modal amplitudes in the case of a speaker-like source. Along with the results of the simulations, a set of rules for implementing acoustic measurements

in a real test rig is given. Finally, the obtained results are employed to calculate the reverberation of an arbitrary signal inside a waveguide.

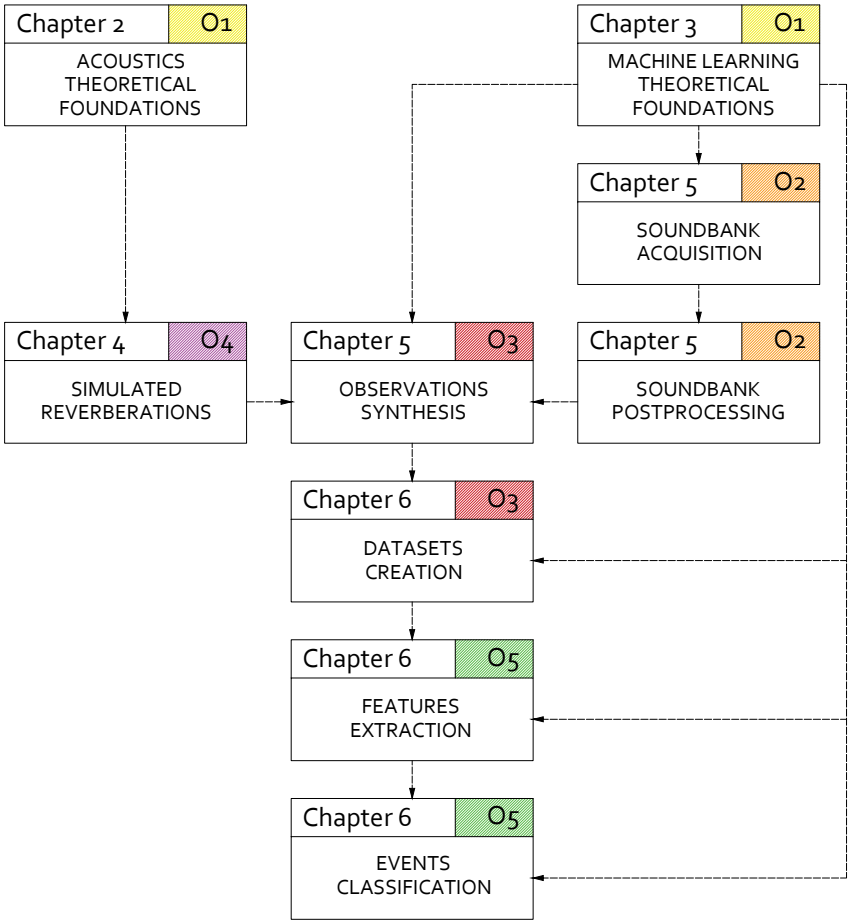


Figure 1.1: *Topic diagram. Related chapters and objectives are reported in the top boxes.*

Chapter 5 describes the approach used to generate collections of strongly-labelled synthetic observations starting from a relatively small set of pipe-related audio recordings. The procedure to acquire the soundbank is illustrated along with the audio filter to remove the background noise and the algorithms to recombine the source signals. The methodology to calculate the simulated reverberations for the generated synthetic observations is derived from the acoustic model. Finally, the structure of the associated annotations is also described.

Chapter 6 defines the details of the dataset synthesis and implements a processing chain for the automatic classification of the acoustic events. A comparison

between several acoustic features is reported along with the techniques adopted to optimise their performance in respect of the classifier used.

The last chapter summarises the results obtained and provides an overview of feature possible developments based on this work.

Chapter 2

Acoustics theory background

This chapter introduces the fundamental notions of acoustics required for this research. The aim is to describe the framework under which the desired model for acoustic propagation in elastic waveguides can be obtained. We focus on straight elastic waveguides filled with an inviscid fluid where an acoustic source with a certain pressure distribution is applied at a section. The choice of this particular configuration is dictated by its similarity to commonly used pipes, the possibility of being modelled using a quasi-analytic approach, and the reproducibility of the simulated results using simple test rigs. As mentioned in section 1.3, the model is a fundamental block of the synthesiser proposed in this work and provides a tool to calculate the synthetic reverberation at a certain point along the pipe. This chapter introduces the necessary theoretical foundations while the actual model is described in chapter 4. The following section provides a brief overview of pertinent literature references and points out this work peculiarities in their respect. Then, the linear elasticity theory is introduced along with the related fundamental wave equations and the analysis of a few relevant configurations for the boundary conditions. Several essential acoustic quantities are also presented. Finally, the characteristic equations for in-pipe propagation are derived for the rigid and the elastic boundary cases. Some results for the rigid boundary case are also generated and reported to better illustrate the effects of the elastic boundaries described in chapter 4.

2.1 Acoustic waveguides: a step beyond the existing literature

Propagation of waves in elastic waveguides has been studied for more than 150 years, and several influential authors, such as Pochhammer, Chree, and Rayleigh, have given their contributions to many specific aspects of the theory [72]. Under the fundamental hypothesis of small mechanical deformations, the theory of *linear elasticity* provides the framework necessary to describe how acoustic events propagate [73]. Pipes are no exception. Acoustic quantities and related elasticity equations are defined taking into account only macroscopic properties of infinitesimal elements of volume containing a sufficiently large number of particles [65, p. 113]. From these equations, from the boundary conditions, and for given sources of acoustic energy, the propagation can be mathematically described. Boundary conditions play a fundamental role: when an acoustic wave crosses a separation boundary between two different mediums, its propagation path is usually altered by reflection and refraction phenomena [88, pp. 1–62]. In waveguides, this process is repeated along the whole domain, and the resulting wave is a combination of rays coming from different paths [88, pp. 63–220].

The study of acoustic propagation in waveguides has found a large variety of applications, from the characterisation of in-pipe communication channels [66],[67] to indirect measurement techniques for the properties of inner medium [89],[76]. Among these, *non-destructive health monitoring* of infrastructures is certainly a field where the characterisation of acoustic waveguides is a fundamental pillar. Some interesting examples are reported by Kundu [68], while many others can be easily found online [90],[91].

2.1.1 Obtaining the impulse response

This work requires calculating the acoustic transformation that a given signal experiences when it propagates through a waveguide. Because of the linearity hypothesis, the problem translates into obtaining the *impulse response* for the acoustic set-up of interest. In general, determining the impulse response is a common issue across many disciplines where linearity is assumed as fundamental

prerequisite [92]. Although equations and measurement techniques depend on the specific matter, the impulse response is usually obtained using one of the following three approaches:

- *empirical*: performing real and specific on-field measurements;
- *analytical*: obtaining the analytical solution of the governing equations;
- *numerical*: using numerical methods to solve the governing equations.

To determine the best approach, it is fundamental to analyse the requirements and focus on the intrinsic benefits and drawbacks of each methodology. Depending on the specific scenario, however, a combination of different solutions can offer a valuable strategy to overcome certain limitations [93].

2.1.1.1 Empirical approaches

The empirical measurement of the impulse response offers the advantage of providing realistic results, but its main limitation is the need to perform the measurements in the actual environment intended to characterise. Although it is possible to find some examples related to in-pipe acoustics [67], this approach is mainly adopted for audio characterisation in architecture [94]. Regardless of the specific implementation, measuring the impulse response generally consists of performing the deconvolution of a reverberated signal recorded in the inspected environment while a certain reference signal is being diffused. Two commonly used techniques are the *maximum length sequence* [95], which uses a pseudo-random noise, and the *exponential sine sweep* [96], which uses a sine signal with an exponential variation of frequency in time. The preferred approach depends on issues such as background noise level, probability of impulsive disturbances, and the maximum length of the recording [97]. Although a direct measurement potentially returns an impulse response close to the real one, several factors can affect the final result. Additive noise and clock mismatch in the acquisition hardware are just two examples. Nevertheless, apart from the difficulties related to the actual signal acquisition, the most relevant issue is generally the intrinsic non-linearity of the measurement equipment [65, pp. 398–416], such as speakers and microphones [98].

2.1.1.2 Analytical approaches

Analytical models can provide the exact theoretical value of the physical quantities under investigation and a fine description of the underlying phenomena, but their applicability is limited to those cases where the geometrical constraints can be matched to specific reference systems, such as cartesian, cylindrical, and spherical. When such a match is possible, the boundary conditions can be generally expressed by simple identities [65, pp.149–160]. For other geometries, analytical models are seldom applicable, but a general analytic understanding of the problem can still offer a tool for qualitative analysis [99].

The acoustic model required in this work concerns the propagation in waveguides of waves generated by certain sources. In linear acoustics, this scenario is mathematically described by inhomogeneous *partial differential equations* (*PDE*) where the source term can be isolated on one side of the identity [100, p. 202]. A typical approach to solving this category of equations consists in separating the source term from the homogeneous equation and focusing first on the latter. Apart from the intrinsic simplification, solving the homogeneous equations offers the extraordinary benefit of providing a general solution whose functions form a complete set, meaning that any function in the same domain can be expressed as a linear combination of the basis obtained from the solution to the homogeneous equation [101, p. 67]. It is important to remark that, although linearly independent, the functions of the basis are not always orthogonal, the latter property being verified only under specific conditions, such as those given by the *Sturm–Liouville theory* [102, pp. 363–366],[103, pp.112–118].

The homogeneous wave equation is a second-order linear PDE in time and space, where, depending on the boundary conditions, some of the independent variables can appear in a coupled form [73, pp. 464–480]. Exploiting the linearity, time derivatives (but also space derivatives) can be avoided by Fourier transformation. Indeed, the transformation of the equation in the frequency domain eliminates the time derivative and turns the homogeneous wave equation into a harmonic Helmholtz equation. Then, because of the superimposition principle, a

set of solutions found in a specific range can be anti-transformed in time to cover wideband signals [104, pp. 299–300]. An approach commonly used to solve the Helmholtz equation is the *separation of the variables* [102, pp. 121–178]. The aim is to turn the homogeneous PDE into a set of independent ordinary differential equations by assuming that the solution can be found as the product of three different functions (each depending on a single space variable). Solutions to the homogeneous equations are either countably or uncountably infinite, the former case being given by bounded domains [101, p.p. 56, 67]. In modal analysis, propagation states corresponding to these solutions (the *eigenfunctions*) provide a decomposition of the overall propagation in different modes, each with peculiar propagation characteristics mainly described in terms of cut-on/off frequencies and phase/group velocities [76].

Once the eigenfunctions have been determined, the modal amplitudes are defined by the time and space characteristics of the driving acoustic source. For the purpose of the impulse response, however, the only concern is the distribution in space. Modal amplitudes are generally calculated by dividing a source distributed in a certain volume into elementary point sources [102, p. 215],[105] and applying the superimposition principle. The harmonic solution of a point source $\delta(\vec{\boldsymbol{x}}-\vec{\boldsymbol{x}}_0)$ placed in $\vec{\boldsymbol{x}}_0$ is generally referred as the Green's function in $\vec{\boldsymbol{x}}_0$. Knowing the Green's function in each point of the source volume allows, through integration, the inference of the harmonic solution for a given source or arbitrary shape [101, pp. 70–76]. The determination of the Green's function is generally not straightforward [106],[107] and additional considerations are reported in the following sections.

For completeness, it is worth mentioning that the modal analysis described above is not the only possible analytic approach and, under certain conditions, adopting other techniques, such as the *method of characteristics* [104, pp. 331–350] and the *method of images* [101, pp. 76–82], can offer more convenient representations of the problem. The method of images, for instance, uses fictitious sources placed outside the domain of interest to avoid the conditions imposed by the boundaries when solving the equations.

2.1.1.3 Numerical approaches

The numerical approach aims to overcome some of the limitations associated with empirical measurements and analytical modelling by using numerical solvers to obtain solutions to the governing equations. Several commercial and open-source software are available for different physical problems [108],[109], some of them sharing the same numerical solver across a variety of different fields [110].

The main benefit of this approach lies in the possibility of calculating the values of the desired physical quantities when analytical representation might be difficult to implement [111]. For instance, complex geometries, variable boundary conditions, and irregular propagation velocities can be treated with numerical models, whereas analytical representations may easily become impracticable [112]. On the other hand, the main drawback is certainly the computational complexity arising even for problems of modest extent at relatively low frequencies [113]. Moreover, since solvers are often employed as black boxes, validating the results can be challenging and errors, such as those caused by poor convergence, can be difficult to discover [114, p. 166–168],[115].

Generally, the key objective of a numerical problem is to solve for a specific field (e.g. displacement field). Other quantities, such as stress and strain, are then derived from the former [114, p. 16]. As for analytical methods, numerical methods can be implemented in the time or frequency domain. Analysis in the time domain provides a direct insight into the transient phenomena regardless of their frequency range but can be affected by several issues, such as stability restrictions on time step, accumulation of error with time, and difficulty in implementing frequency-dependent boundary conditions [112]. Alternatively, the impulse response can be obtained from the inverse Fourier transform of the solutions in the frequency domain. This approach, however, can yield non-causal effects linked to the "wraparound" of the discrete Fourier transform and can suffer dispersion-related phase error [112]. Although a range of different algorithms populates the family of numeric solvers, two groups of implementations are particularly relevant in terms of popularity [101],[116]: *finite differences* (*FD*) and *finite elements* (*FE*).

Finite difference method

The finite difference method aims to approximate the governing equations by replacing the derivatives with difference quotients and calculating the numeric values in a discrete lattice of points. The function describing the field of a certain acoustic quantity is expanded in Taylor's series around the point of the lattice where the derivative is supposed to be calculated. Then, simple manipulations return the value of the derivative, which is written as the sum of its approximation plus an error depending on the step of the lattice [117, pp. 19–20]. Once the discrete derivatives are obtained, the given problem turns into a matrix equation $\boldsymbol{AX} = \boldsymbol{B}$, where $\boldsymbol{X}$ are the unknown field values, $\boldsymbol{B}$ the source components, and $\boldsymbol{A}$ a matrix that accounts for the derivatives and the rest of the environmental parameters. All the terms are evaluated at the points of the lattice [101, p. 177].

The simplest form of approximation for the first derivative can be obtained using only one neighbouring point (*forward difference* or *backward difference*). When solving the equations, however, it is fundamental to minimise the approximation error and, apart from increasing the lattice density, a workaround consists in using higher order schemes where derivatives are calculated using more neighbouring points [118, pp. 18–21]. For instance, the simple *central difference approximation* calculates the first derivative by accounting for the two closest points (for a function of a single variable). Unfortunately, this artifice also requires a grid of points equally spaced, and since the lattice cannot be selectively refined where geometries are more complex, this issue remains one of the main limitations of FD methods. Besides, geometrical models with complex boundaries crossing the lattice in oblique directions produce an unavoidable "staircase" effect and a coarse approximation of the boundary conditions [116].

Although several improvements have been proposed (for example, the *Local Interaction Simulation Approach* allows better characterisation of the boundary conditions [117, pp. 33–36], while *Finite Integration Technique* discretises the integral rather than the differential equations [119]), the main limitation remains the necessity to adopt an evenly spaced lattice. Indeed, the number of points in $\boldsymbol{X}$ grows as n^3, where n is the chosen number of points per wavelength [101, p.196].

Finite element method

To overcome the issues above, software applications often adopt finite element solvers. Commercial algorithms are generally protected by intellectual property, but some solutions are reported in the literature [120] along with examples of open-source software [121],[109].

In the FE paradigm, the domain is divided into a *mesh* of small geometric elements such as hexahedra, wedges, and tetrahedra [122, pp.24–26] and the solution inside the elements is approximated as a linear combination of a set of *shape functions* in a predefined basis [101, pp. 216–218]. The main benefit of the finite element approach is the flexibility achievable for the discretisation of the domain. Although different elements require different mathematical formulations [114, pp. 187–241], there is no strict requirement on the geometrical lattice as for the FD case. Indeed, the mesh can be refined using a higher number of elements only where geometries are more intricate. Besides, since the position of the nodes can be arbitrarily defined, the staircase problem on the boundaries is largely mitigated. It is important to note, however, that the quality of the final solution depends on the elements chosen. For instance, triangular or tetrahedral elements (especially when stretched along a particular direction) generally yield worse results than brick/hexahedral elements, but the former allow better discretisation of complex geometries [114, pp. 180–181].

The implementation of the actual finite element solver generally follows one of the two following methods: *variational* and *weighted residuals* [117, pp. 19–20]. The variational method exploits the energy balance between total internal energy and external work [116]. Solutions correspond to the minimum mechanical energy found from the relationship between internal parameters, such as stress, strain, displacement, and externally applied inputs and constraints [114, pp. 187–241]. On the other hand, the weighted residuals method imposes a trial function built from the basis as a solution to the differential equation. Since this function is generally not a true solution, the evaluation of the governing equations returns an error called *residual*. The residual is then minimised to refine the trial function and find a solution as close as possible to the real one [117]. Using the *Galerkin's method*, for

instance, the approximation is optimised by imposing the orthogonality between the residual and the shape functions [101, pp. 216–218].

Despite the benefits, adopting FEM simulators hides several important issues. For instance, the termination of the discretised region requires special elements [122, pp. 26–30] whose choice and position must be carefully assessed. A numerical phase shift can accumulate at each element and introduce a relevant phase error, especially at higher frequencies (*pollution effect*) [112]. Higher space and time resolution can be used for better approximations, but the computational complexity can grow quickly beyond the available resources in common calculators [116]. Although the coarsest mesh that guarantees convergence can be used, such a solution is not known in advance, and its determination might not be trivial [114, p. 176]. Finally, the cost of commercial software can be a substantial barrier.

2.1.2 A model for the synthesiser

Several issues have been considered to identify the best approach for the present work. Firstly, the lack of access to a variety of water infrastructures and the impossibility of acquiring a sufficient number of real recordings makes the empirical determination of the impulse response a poor option. Solving the propagation problem using numerical techniques, such as frequency domain FE, is certainly a better alternative; however, albeit FE allows the flexibility to account for arbitrary geometries and sources, it also introduces a few significant drawbacks. Even omitting the cost of specialised software (warranty-less open-source alternatives are available), obtaining correct numerical solutions requires a proper configuration of the model itself and a check of the coherence of the results. For instance, as seen in the previous section, the mesh (the distribution of the elements and their density) and the boundary conditions (both real and fictitious) should be defined before running the model. Although some operations can be automatically performed, other tasks require manual activities and supervision with a clear understanding of the physical problem. Besides, the model needs solving for the maximum size, and large 3D domains can be computationally intensive, even beyond the limits of standard laptops. The last of the three options

mentioned above, the analytical approach, is also very limited. Indeed, apart from elementary geometries, it is generally challenging to obtain analytic solutions for the governing equations. Finally, the model developed is intended to be used in a synthesiser where, during the synthesis, calculations are repeated multiple times. Ideally, the simulator adds only a little extra computational cost per iteration. Moreover, although accounting for a large variety of geometries and sources could be beneficial, the aim remains understanding how reverberation affects in-pipe acoustic events. From this point of view, a simulator for the most common pipe geometries and materials, possibly matched to simple real test rigs, is a sufficient starting point.

2.1.2.1 Quasi-analytic calculation of the impulse response

Considering the constraints above, combining numerical and analytical techniques can offer a better solution. This approach is generally referred to as *semi-analytic finite elements* (*SAFE*) [116] and aims to use an analytic formulation along those directions where a numeric approach is unnecessary. Along the other directions, the domain is discretised with reduced dimensionality achieving a substantial reduction of the computational complexity. In straight waveguides, for instance, the numerical analysis can be limited to 2D cross-sections using fewer and simpler elements. The FE eigenproblem yields the axial wavenumbers $k_z(\omega)$, and the solution in the propagation direction z can be obtained in the analytical form [123].

SAFE applications have a few decades of records in the literature [124], and, apart from the reduced computational complexity, their success in waveguide analysis is also related to the possibility of accounting for arbitrary cross-section geometries [125], combinations of different materials [126], and even viscoelastic damping [127]. SAFE methods, however, retain part of the issues related to full FE techniques. For instance, it is necessary to check for convergence, sometimes using a finer mesh to compensate for the post-processing loss of accuracy [127]. In [126], the convergence is verified by solving the discrete domain multiple times until an acceptable match between theoretical and approximated cut-on/cut-off

frequency is found.

Although SAFE methods are a viable option for this work, the cylindrical geometry of the pipes suggests that an analytical approach can further replace some numerical aspects of the simulations. Indeed, it is possible to find analytic solutions to the homogeneous equation for multilayered cylinders [76] (even accounting for viscoelastic dumping [89]), but the integration of the source requires additional considerations. Using the separation of variables mentioned in section 2.1.1.2, the problem turns into a search for the eigenvalues on the waveguide cross-section. Once the axial wavenumbers $k_z(\omega)$ are obtained, to define a unique solution, it is necessary to calculate the modal amplitudes from the source excitation.

This problem is generally tackled by decomposing the source as a sum of elementary sources and using Green's functions along with the superimposition principle (section 2.1.1.2). Importantly, the orthogonality is the underlying fundamental assumption for a precise calculation of the amplitudes [101, pp. 70–76]. Indeed, Green's functions relate to point sources, and their exact approximation requires an infinite number of modes. If modes are orthogonal, it is possible to calculate their exact amplitude independently from the others, and the approximation error depends only on the truncation of the series. This property is no longer valid if the modes are not orthogonal. In fact, since only a small number of modes can be accounted for, the truncation of a series of non-orthogonal modes would also introduce an error for the amplitude of the modes retained. Therefore, the determination of Green's functions is not a viable option in the case of non-orthogonal modes.

Orthogonality provides significant simplifications but cannot always be assumed verified. In [128], the proof of orthogonality for a multilayered waveguide of an arbitrary cross-section is given from real (or complex) reciprocity [88, pp. 151–155]. The demonstration, however, requires free or rigid boundary conditions, continuity of the normal stress, and continuity of displacement across the separation boundaries [129]. Although free boundary conditions for the outermost surface can be here accepted, the required liquid-solid interface imposes only continuity of normal displacement, and this condition is not sufficient to guarantee the proof of orthog-

onality reported in [128]. Therefore, as shown in chapter 4, two different novel solutions are proposed to determine the modal amplitudes directly from the source distribution. Moreover, since mode sorting algorithms exploiting orthogonality [123, p. 140] cannot be applied, a different novel algorithm is also described.

The approach adopted in this work is defined as *quasi-analytic* because, apart from the discretisation of the pressure source, no other numeric modelling is involved. The main limitations are the need to place the source on the cross-section of the waveguide and, as for any analytic approach, the possibility of dealing with geometries that cannot be matched with the reference system. These issues, however, can be partially overcome by mixing the proposed solution with FE techniques. Sources of arbitrary geometries, for instance, could be virtualised by obtaining the equivalent pressure or displacement on a 2D interface [130], while obstacles or elements with complex shapes can be modelled with FE techniques and interfaced with region solved using the analytical approach [131],[132]. In this work, however, these aspects are not further developed.

In the following sections, some necessary concepts of linear elasticity are introduced. The analysis of the boundary conditions is restricted to the cases required for the model developed. The study of a simple cylindrical waveguide with rigid boundary conditions is reported along with some simulated results to explain the role of orthogonality in calculating the modal amplitudes. Finally, the governing equations for an elastic waveguide filled with an inviscid liquid are introduced and manipulated to obtain the characteristic equation for the calculation of the wavenumbers. The novel model proposed is discussed in chapter 4.

2.2 Fundamental concepts of linear elasticity

The displacement of a particle in an elastic body around its equilibrium position is usually referred to as *vibration*. The fundamental theory behind vibrations describes the relations between forces and deformations in solid bodies and how the effects of certain mechanical events propagate along the body itself. As shown in the following sections, many considerations valid for propagation in fluids can be regarded as a particular case of propagation in solids.

2.2.1 Stress and strain

Two physical quantities are fundamental for the vibration theory: *stress* and *strain*. An accurate description can be found in many books and lecture notes of mechanics [74, pp. 31–48, 55–71],[133, pp. 29–112]. Here only the necessary properties related to isotropic media are recalled.

2.2.1.1 Stress

The definition of stress is related to the definition of *traction vector*. Let S_z be a flat surface centred in P, with P a point in a solid body. Let $\vec{\boldsymbol{x}}$, $\vec{\boldsymbol{y}}$, $\vec{\boldsymbol{z}}$ be a set of orthonormal versors centered in P with $\vec{\boldsymbol{z}}$ normal to the surface S_z, and $\vec{\boldsymbol{F}}_z$ the force acting on the latter. The traction vector can be defined as:

$$\vec{\boldsymbol{t}}^{(z)} = \lim_{S_z \to 0} \frac{\vec{\boldsymbol{F}}_z}{S_z}. \tag{2.1}$$

A specific instance of the traction vector is identified when P, the direction of normal unit vector $\vec{\boldsymbol{z}}$, and the force $\vec{\boldsymbol{F}}_z$ are specified. In the reference system identified by $\vec{\boldsymbol{x}}$, $\vec{\boldsymbol{y}}$ and $\vec{\boldsymbol{z}}$, the traction has a component normal to the surface (along $\vec{\boldsymbol{z}}$) and two others parallel to S_z (along $\vec{\boldsymbol{x}}$ and $\vec{\boldsymbol{y}}$). With the same P and the same reference system $\vec{\boldsymbol{x}}\vec{\boldsymbol{y}}\vec{\boldsymbol{z}}$, similar definitions can be given for surfaces normal to $\vec{\boldsymbol{x}}$ and $\vec{\boldsymbol{y}}$. Therefore, given a point P and a reference system associated with it, it is possible to define nine scalar quantities σ_{ij} from the components of the related traction vectors. For $i = j$ these quantities are called *normal stress*, while, in the other cases, they are called *shear stress* and sometimes are indicated with the Greek letter τ. The first index indicates the direction of the normal, while the second index indicates the component of the stress. For instance, σ_{xx} indicates the normal stress along the x direction. Since traction is the force per unit area exerted by the material above the surface on the material below the surface, Newton's third law entails that an equal and opposite traction must be exerted in the opposite direction. In particular, if an infinitesimal cube of material is considered around the point P, the stresses on opposite sides of the cube tend to be equal and opposite.

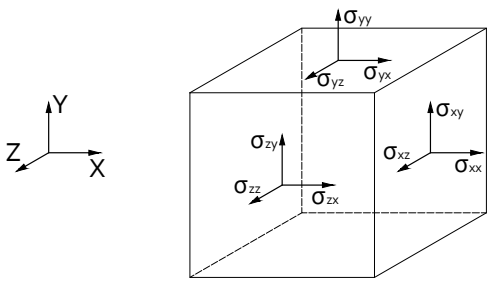


Figure 2.1: *Stress components for an infinitesimal cube of material.*

Figure 2.1 illustrates the definitions of the stress components σ_{ij}, which can also be organised in a matrix form:

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix}, \tag{2.2}$$

where the matrix $\boldsymbol{\sigma}$ is called *stress tensor*. The specific form of the stress tensor is related to the specific coordinate reference system, but named $\boldsymbol{Q}$ the rotation matrix to transform the reference coordinate system $\vec{\boldsymbol{x}}\,\vec{\boldsymbol{y}}\,\vec{\boldsymbol{z}}$ into $\vec{\boldsymbol{x}}'\,\vec{\boldsymbol{y}}'\,\vec{\boldsymbol{z}}'$[1], the relationship between the expression of the stress tensor in the first coordinate system σ and the one in the second coordinate system σ' can be written as [74, p. 60]:

$$\boldsymbol{\sigma} = \boldsymbol{Q}\boldsymbol{\sigma}'\boldsymbol{Q}^T \qquad \boldsymbol{\sigma}' = \boldsymbol{Q}^T\boldsymbol{\sigma}\boldsymbol{Q}, \tag{2.3}$$

where superscript ^{T} indicates the matrix transpose. Interestingly, for a given stress tensor, there exists a specific reference system where the values of the shear stresses are zero, and the normal stresses include both the maximum and the minimum values. The directions of this particular reference system are called *principal directions*, and the associated stresses are called *principal stresses*. It can be proved that the principal stresses are the eigenvalues of the stress tensor,

[1] the rotation matrix is built with the director cosines of the related axis.

while the principal directions are the related eigenvectors [134, pp. 209–211]. On the other hand, the shear stresses reach the maximum value on planes oriented 45° from the principal directions, and they have magnitudes equal to half of the difference between the related principal stresses. Regardless of the specific representation, for isotropic materials, the stress tensor is always symmetric. That is, $\sigma_{ij} = \sigma_{ji}$ and three *invariants* can be defined [74, p. 61]:

$$I_{\sigma 1} = \sigma_{xx} + \sigma_{yy} + \sigma_{zz} \tag{2.4a}$$

$$I_{\sigma 2} = \sigma_{xx}\sigma_{yy} + \sigma_{yy}\sigma_{zz} + \sigma_{zz}\sigma_{xx} - \sigma_{xy}^2 - \sigma_{yz}^2 - \sigma_{zx}^2 \tag{2.4b}$$

$$I_{\sigma 3} = \sigma_{xx}\sigma_{yy}\sigma_{zz} - \sigma_{xx}\sigma_{yz}^2 - \sigma_{yy}\sigma_{zx}^2 - \sigma_{zz}\sigma_{xy}^2 + 2\sigma_{xy}\sigma_{yz}\sigma_{zx} \tag{2.4c}$$

The stress tensor offers a useful property stated by *Cauchy's law* [134, p. 203]. Specifically, given a surface whose normal unit vector is $\vec{\boldsymbol{n}}$, the related traction can be calculated as:

$$\vec{\boldsymbol{t}} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix} \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} = \boldsymbol{\sigma}\vec{\boldsymbol{n}} \tag{2.5}$$

where the components of the traction, the tensor, and the normal surface vector are all given in the same reference coordinate system. It is worth remarking that when the stress tensor can be expressed as:

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_0 & 0 & 0 \\ 0 & \sigma_0 & 0 \\ 0 & 0 & \sigma_0 \end{bmatrix} \tag{2.6}$$

the stress is referred as *isotropic state of stress* or *hydrostatic state of stress* [74, p. 84], and the tensor is invariant. This condition is verified for a fluid at rest with pressure σ_0.

2.2.1.2 Strain

When a set of forces acts upon a material, the shape of the latter changes. Regardless of the causes, there exist several possible ways to measure the deformation, and the best approach depends on the scope of the analysis. All the different definitions [74, p. 113], however, exhibit negligible differences when variations of the shape are small. In this case, a simple linear approach provides good accuracy and simple calculations.

Intuitively, a solid body is subject to deformation in a point P when virtual segments passing through P and connecting P to other points of the solid body stretch, contract, or rotate. It can be proved that the deformation at a point P (figure 2.2) can be completely characterised if the behaviour of a set of three perpendicular segments passing through P is known [133, p. 91].

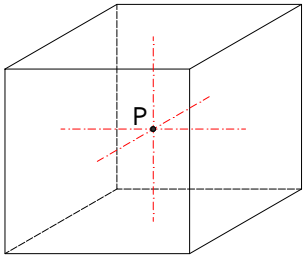


Figure 2.2: *Deformation in a point P. Virtual segments passing through P and connecting P to other points of the solid body give a measure of the deformation.*

In this work, the deformations are measured according to the concept of *Engineering* or *small strain* [74, p. 34–38], which provides both linearity and accuracy for small deformations. To introduce the concept, it is assumed a 3D coordinate reference system $\vec{\boldsymbol{x}}\vec{\boldsymbol{y}}\vec{\boldsymbol{z}}$ and the 3D deformation analysis is decomposed into three 2D analysis along the main planes of the reference system. From a point in the solid body $P \equiv x, y, z$ (consider now its 2D representation $P \equiv x, y$ on a plane $z = constant$), two segments can be drawn in the direction of $\vec{\boldsymbol{x}}$ and $\vec{\boldsymbol{y}}$: $\overline{PA}$ and $\overline{PB}$ (figure **??**). If the solid body changes its shape, these two segments move in space

and the new positions of P, A and B are now P', A' and B'. For small deformations, the segment $\overline{P'A'}$ is almost the same length as $\overline{P'A'_x}$. The *engineering normal strain* along $\vec{\boldsymbol{x}}$ is defined as:

$$\varepsilon_{xx} = \frac{\overline{P'A'_x} - \overline{PA}}{\overline{PA}} \tag{2.7}$$

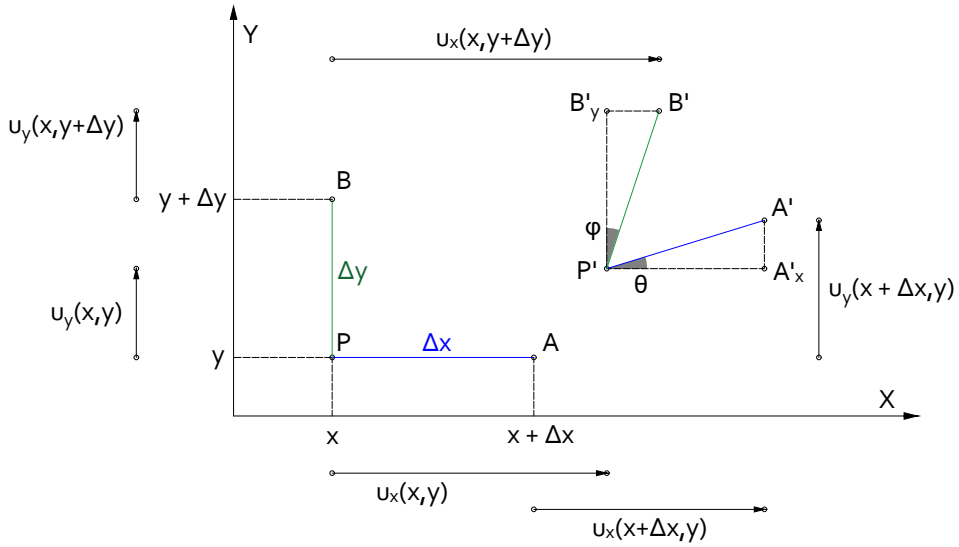


Figure 2.3: *Representation of the strain at a point P in two dimensions.*

The position of P' and A'_x in respect of P and A can be expressed using the x component of the *displacement vector* $\vec{\boldsymbol{u}}$. Therefore, the normal strain can be rewritten as:

$$\varepsilon_{xx} = \lim_{\Delta x \to 0} \frac{u_x(x+\Delta x, y) - u_x(x,y)}{\Delta x} = \frac{\partial u_x}{\partial x}. \tag{2.8}$$

Repeating the same procedure in the $\vec{\boldsymbol{y}}$ direction:

$$\varepsilon_{yy} = \frac{\partial u_y}{\partial y}. \tag{2.9}$$

Hence, the normal engineering strain can be written in terms of the partial derivative of the displacement $\vec{\boldsymbol{u}}$. To include all the effects of the deformation, also the angles θ and ϕ must be considered. For small deformations, it is possible to approximate θ and ϕ with their respective tangents, thus:

$$\theta \approx \frac{\overline{A'_x A'}}{\overline{P' A'_x}} \qquad \phi \approx \frac{\overline{B'_y B'}}{\overline{P' B'_y}}, \tag{2.10}$$

and it can be assumed $\overline{P'A'_x} \approx \overline{PA}$ and $\overline{P'B'_y} \approx \overline{PB}$. The *engineering shear strain* is then defined as:

$$\varepsilon_{xy} = \frac{1}{2}\Big(\frac{\overline{A'_x A'}}{\overline{PA}} + \frac{\overline{B'_y B'}}{\overline{PB}}\Big) \tag{2.11}$$

and, using the displacement vector $\vec{\boldsymbol{u}}$ for small values of Δx and Δy:

$$\varepsilon_{xy} = \frac{1}{2}\Big(\lim_{\Delta x \to 0} \frac{u_y(x+\Delta x, y) - u_y(x,y)}{\Delta x} + \lim_{\Delta y \to 0} \frac{u_x(x, y+\Delta y) - u_x(x,y)}{\Delta y}\Big) = \frac{1}{2}\Big(\frac{\partial u_y}{\partial x} + \frac{\partial u_x}{\partial y}\Big). \tag{2.12}$$

The definitions above can be repeated for the planes xz and yz, and the strains can be organised in a symmetric matrix form. The *strain tensor* in cartesian and cylindrical coordinates is therefore defined as [74, p. 47]:

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \frac{\partial u_x}{\partial x} & \frac{1}{2}\Big(\frac{\partial u_y}{\partial x} + \frac{\partial u_x}{\partial y}\Big) & \frac{1}{2}\Big(\frac{\partial u_z}{\partial x} + \frac{\partial u_x}{\partial z}\Big) \\ \frac{1}{2}\Big(\frac{\partial u_y}{\partial x} + \frac{\partial u_x}{\partial y}\Big) & \frac{\partial u_y}{\partial y} & \frac{1}{2}\Big(\frac{\partial u_y}{\partial z} + \frac{\partial u_z}{\partial y}\Big) \\ \frac{1}{2}\Big(\frac{\partial u_z}{\partial x} + \frac{\partial u_x}{\partial z}\Big) & \frac{1}{2}\Big(\frac{\partial u_y}{\partial z} + \frac{\partial u_z}{\partial y}\Big) & \frac{\partial u_z}{\partial z} \end{bmatrix} \tag{2.13}$$

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \frac{\partial u_r}{\partial r} & \frac{1}{2}\Big(\frac{1}{r}\frac{\partial u_r}{\partial \theta} + \frac{\partial u_\theta}{\partial r} - \frac{u_\theta}{r}\Big) & \frac{1}{2}\Big(\frac{\partial u_r}{\partial z} + \frac{\partial u_z}{\partial r}\Big) \\ \frac{1}{2}\Big(\frac{1}{r}\frac{\partial u_r}{\partial \theta} + \frac{\partial u_\theta}{\partial r} - \frac{u_\theta}{r}\Big) & \frac{1}{r}\Big(u_r + \frac{\partial u_\theta}{\partial \theta}\Big) & \frac{1}{2}\Big(\frac{\partial u_\theta}{\partial z} + \frac{1}{r}\frac{\partial u_z}{\partial \theta}\Big) \\ \frac{1}{2}\Big(\frac{\partial u_r}{\partial z} + \frac{\partial u_z}{\partial r}\Big) & \frac{1}{2}\Big(\frac{\partial u_\theta}{\partial z} + \frac{1}{r}\frac{\partial u_z}{\partial \theta}\Big) & \frac{\partial u_z}{\partial z} \end{bmatrix} \tag{2.14}$$

Many of the properties of the stress tensor have an equivalent for the strain. For instance, with similar calculations, it is possible to find the *principal strains* (which include both minimum and maximum) and the related *principal directions*, that is, the spatial directions of the principal strains [74, p. 44]. As for the stress, the shear strain is null along the principal directions and reaches its maximum value on planes oriented 45° from the principal directions. The strain tensor admits three *invariants*, and the definition can be obtained from (2.4) substituting σ with ε. In this case, the first invariant $I\varepsilon_1$ also has a geometric meaning since:

$$I\varepsilon_1 = \varepsilon_{xx} + \varepsilon_{yy} + \varepsilon_{zz} \approx \frac{\Delta V}{V}, \tag{2.15}$$

where V is the volume to which the strain is referred to, and ΔV is its variation.

2.2.2 Linear elasticity

Linear elastic models are based on the following assumptions [74, p. 77]:

- stress and strain are linked by a unique linear relationship;
- stress-strain relationship is time-independent;
- stress does not induce any permanent deformation.

Along with the conditions above, it is assumed that the material is *isotropic*, that is, with the same mechanical properties along any space direction [133, p. 113]. The linear elastic behaviour of the materials is commonly characterised using the parameters described below.

The *Young's modulus* (Fig. 2.4-A) characterises the relationship between stress and strain when a cylindrical specimen of material is stretched under a tensile test. It measures the deformation Δl in the linear region (before the *elastic limit*) in the same direction of the applied stretching force:

$$E = \frac{\sigma_l}{\Delta l / l_0} = \frac{\sigma_l}{\varepsilon_l}. \tag{2.16}$$

Equation (2.16) is also called the one dimensional linear elastic *constitutive*

equation. The unit of Young's modulus is Pascal $[Pa]$.

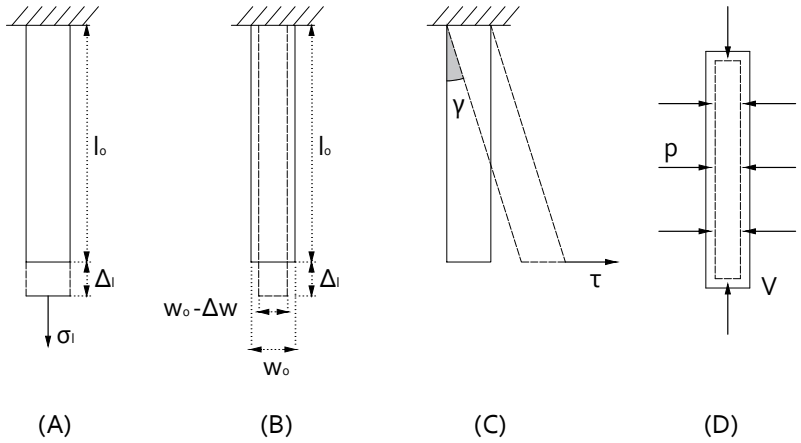


Figure 2.4: *Representation of the elasticity constants. A) Young's modulus, B) Poisson's ratio, C) Shear modulus, D) Bulk modulus*

Under the same conditions described for Young's modulus, the *Poisson's ratio* (figure 2.4-B) measures how the specimen changes its thickness with respect to the deformation along the direction of the force:

$$\nu = -\frac{\Delta w/w_0}{\Delta l/l_0} = -\frac{\varepsilon_w}{\varepsilon_l}, \tag{2.17}$$

where w_0 is the original diameter of the cylindrical specimen, and Δw is its variation. The Poisson's ratio is dimensionless. The *shear modulus* (figure 2.4-C) is the equivalent of Young's modulus for the shear stress and the shear strain, and it is defined as:

$$\mu = \frac{\tau}{\gamma} = \frac{\tau}{2\varepsilon}. \tag{2.18}$$

The unit of the shear modulus is $[Pa]$. The *bulk modulus* (figure 2.4-D) measures the change of volume of a specimen undergoing an equal pressure on all its sides:

$$K = -\frac{p}{\Delta V/V}. \tag{2.19}$$

The unit of the bulk modulus is $[Pa]$.

The parameters introduced above can be combined to find the *constitutive equations*, that is, the relationships between strain and stress in space. It is important to note that the linearity hypothesis allows the use of the *superimposition principle*, while restricting the analysis to isotropic materials allows the exploitation of symmetries. From (2.16) and (2.17), the stress along $\vec{\boldsymbol{x}}$ induced by the normal stresses can be written as:

$$\varepsilon_{xx}^{x} = \frac{1}{E}\sigma_{xx} \qquad \varepsilon_{xx}^{y} = -\frac{\nu}{E}\sigma_{yy} \qquad \varepsilon_{xx}^{z} = -\frac{\nu}{E}\sigma_{zz}, \tag{2.20}$$

where the isotropic assumption entails that E and ν are equal along all the directions and that no shear stress along $\vec{\boldsymbol{x}}$, $\vec{\boldsymbol{y}}$, $\vec{\boldsymbol{z}}$ can arise from normal stress along $\vec{\boldsymbol{x}}$, $\vec{\boldsymbol{y}}$, $\vec{\boldsymbol{z}}$. Therefore:

$$\varepsilon_{xx} = \frac{1}{E}\Big(\sigma_{xx} - \nu(\sigma_{yy} + \sigma zz)\Big) \tag{2.21a}$$

$$\varepsilon_{yy} = \frac{1}{E}\Big(\sigma_{yy} - \nu(\sigma_{xx} + \sigma zz)\Big) \tag{2.21b}$$

$$\varepsilon_{zz} = \frac{1}{E}\Big(\sigma_{zz} - \nu(\sigma_{yy} + \sigma zz)\Big). \tag{2.21c}$$

It can be proved [133, p. 147] that, for an isotropic linear elastic material, the following relationship between E, ν, μ exists:

$$\mu = \frac{E}{2(1+\nu)}. \tag{2.22}$$

Combining (2.22) with (2.18) yields:

$$\varepsilon_{xy} = \frac{1+\nu}{E}\sigma_{xy} \tag{2.23a}$$

$$\varepsilon_{yz} = \frac{1+\nu}{E}\sigma_{yz} \tag{2.23b}$$

$$\varepsilon_{zx} = \frac{1+\nu}{E}\sigma_{zx}. \tag{2.23c}$$

Equations (2.21) and (2.23) are usually known as *Hooke's law* and are the

constitutive equations for a linear isotropic material. For calculation purposes, Hooke's law is usually given in matrix form:

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{xy} \\ \sigma_{yz} \\ \sigma_{zx} \end{bmatrix} = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 2\mu & 0 & 0 \\ 0 & 0 & 0 & 0 & 2\mu & 0 \\ 0 & 0 & 0 & 0 & 0 & 2\mu \end{bmatrix} \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \varepsilon_{xy} \\ \varepsilon_{yz} \\ \varepsilon_{zx} \end{bmatrix} = \boldsymbol{H\varepsilon}, \quad (2.24)$$

where:

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \qquad \mu = \frac{E}{2(1+\nu)} \quad (2.25)$$

are called *Lamé constants*. Hooke's law in cylindrical coordinates can be written as [74, p. 86]:

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{rr} \\ \sigma_{\theta\theta} \\ \sigma_{zz} \\ \sigma_{r\theta} \\ \sigma_{\theta z} \\ \sigma_{zr} \end{bmatrix} = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 2\mu & 0 & 0 \\ 0 & 0 & 0 & 0 & 2\mu & 0 \\ 0 & 0 & 0 & 0 & 0 & 2\mu \end{bmatrix} \begin{bmatrix} \varepsilon_{rr} \\ \varepsilon_{\theta\theta} \\ \varepsilon_{zz} \\ \varepsilon_{r\theta} \\ \varepsilon_{\theta z} \\ \varepsilon_{zr} \end{bmatrix} = \boldsymbol{H\varepsilon}. \quad (2.26)$$

Combining (2.13) and (2.24) together, it is possible to write the stress as a function of the displacement:

$$\begin{aligned} \sigma_{xx} &= (\lambda+2\mu)\frac{\partial u_x}{\partial x} + \lambda\frac{\partial u_y}{\partial y} + \lambda\frac{\partial u_z}{\partial z} & \sigma_{xy} &= \mu\left(\frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x}\right) \\ \sigma_{yy} &= \lambda\frac{\partial u_x}{\partial x} + (\lambda+2\mu)\frac{\partial u_y}{\partial y} + \lambda\frac{\partial u_z}{\partial z} & \sigma_{yz} &= \mu\left(\frac{\partial u_y}{\partial z} + \frac{\partial u_z}{\partial y}\right) \\ \sigma_{zz} &= \lambda\frac{\partial u_x}{\partial x} + \lambda\frac{\partial u_y}{\partial y} + (\lambda+2\mu)\frac{\partial u_z}{\partial z} & \sigma_{zx} &= \mu\left(\frac{\partial u_z}{\partial x} + \frac{\partial u_x}{\partial z}\right) \end{aligned} \quad (2.27)$$

and, equivalently, from (2.14) and (2.26) in cylindrical coordinates:

$$\begin{aligned}
\sigma_{rr} &= (\lambda+2\mu)\frac{\partial u_r}{\partial r} + \frac{\lambda}{r}\left(\frac{\partial u_\theta}{\partial \theta} + u_r\right) + \lambda\frac{\partial u_z}{\partial z} & \sigma_{r\theta} &= \mu\left(\frac{1}{r}\frac{\partial u_r}{\partial \theta} + \frac{\partial u_\theta}{\partial r} - \frac{u_\theta}{r}\right) \\
\sigma_{\theta\theta} &= \lambda\frac{\partial u_r}{\partial r} + \frac{\lambda+2\mu}{r}\left(\frac{\partial u_\theta}{\partial \theta} + u_r\right) + \lambda\frac{\partial u_z}{\partial z} & \sigma_{\theta z} &= \mu\left(\frac{\partial u_\theta}{\partial z} + \frac{1}{r}\frac{\partial u_z}{\partial \theta}\right) \\
\sigma_{zz} &= \lambda\frac{\partial u_r}{\partial r} + \frac{\lambda}{r}\left(\frac{\partial u_\theta}{\partial \theta} + u_r\right) + (\lambda+2\mu)\frac{\partial u_z}{\partial z} & \sigma_{zr} &= \mu\left(\frac{\partial u_r}{\partial z} + \frac{\partial u_z}{\partial r}\right).
\end{aligned} \tag{2.28}$$

2.2.3 The Navier's Equation

The definition of stress given in the previous section can be used in combination with Newton's second law to obtain the *equation of motion*. Given an infinitesimal cube of material centred in $P \equiv (\bar{x}, \bar{y}, \bar{z})$, of dimension $\Delta x \Delta y \Delta z$ and density ρ, the forces acting on it can be divided in body forces (such as gravity) and stress (such as pressure). Since the dimension is infinitesimal, the density ρ and the body forces $\vec{\boldsymbol{b}}$ can be considered constant, while the stress can be considered linearly variable from one face of the cube to the opposite. Along the $\vec{\boldsymbol{x}}$ direction, for example, named $\sigma_{xx}(\bar{x} - \Delta x/2)$ the average normal stress on the face $x = \bar{x} - \Delta x/2$, the stress on the face $x = \bar{x} + \Delta x/2$ can be obtained using Taylor's series up to the first derivative (since the cube is infinitesimal and the stress variation can be assumed linear, the error is negligible). Thus, the net force associated with the normal stress acting on the cube in the $\vec{\boldsymbol{x}}$ direction is:

$$F_{xx} = \left(\sigma_{xx}(\bar{x} - \Delta x/2) + \frac{\partial \sigma_{xx}}{\partial x}\Delta x - \sigma_{xx}(\bar{x} - \Delta x/2)\right)\Delta y \Delta z = \frac{\partial \sigma_{xx}}{\partial x}\Delta x \Delta y \Delta z \tag{2.29}$$

Since the stress variation is assumed linear, the derivative is constant along $\vec{\boldsymbol{x}}$ and it can be calculated in P. Similar steps can be repeated for the shear stress, and Newton's second law along $\vec{\boldsymbol{x}}$ can be written as:

$$\left.\frac{\partial \sigma_{xx}}{\partial x}\right|_P \Delta x \Delta y \Delta z + \left.\frac{\partial \sigma_{xy}}{\partial y}\right|_P \Delta y \Delta x \Delta z + \left.\frac{\partial \sigma_{xz}}{\partial z}\right|_P \Delta z \Delta x \Delta y + b_x = \rho \Delta x \Delta y \Delta z \, a_x. \tag{2.30}$$

Repeating the same procedure along $\vec{\boldsymbol{y}}$ and $\vec{\boldsymbol{z}}$ yields the *equations of motion*:

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z} + b_x = \rho a_x \tag{2.31a}$$

$$\frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{yx}}{\partial x} + \frac{\partial \sigma_{yz}}{\partial z} + b_y = \rho a_y \tag{2.31b}$$

$$\frac{\partial \sigma_{zz}}{\partial z} + \frac{\partial \sigma_{zx}}{\partial x} + \frac{\partial \sigma_{zy}}{\partial y} + b_z = \rho a_z. \tag{2.31c}$$

Combining (2.31) with (2.24) and (2.13) the *Navier's equations* are obtained:

$$\rho\, a_x = \tag{2.32a}$$
$$b_x + (\lambda + 2\mu)\frac{\partial^2 u_x}{\partial x^2} + \mu\frac{\partial^2 u_x}{\partial y^2} + \mu\frac{\partial^2 u_x}{\partial z^2} + (\lambda + \mu)\frac{\partial^2 u_y}{\partial x \partial y} + (\lambda + \mu)\frac{\partial^2 u_z}{\partial x \partial z}$$

$$\rho\, a_y = \tag{2.32b}$$
$$b_y + (\lambda + 2\mu)\frac{\partial^2 u_y}{\partial y^2} + \mu\frac{\partial^2 u_y}{\partial x^2} + \mu\frac{\partial^2 u_y}{\partial z^2} + (\lambda + \mu)\frac{\partial^2 u_x}{\partial y \partial x} + (\lambda + \mu)\frac{\partial^2 u_z}{\partial y \partial z}$$

$$\rho\, a_z = \tag{2.32c}$$
$$b_z + (\lambda + 2\mu)\frac{\partial^2 u_z}{\partial z^2} + \mu\frac{\partial^2 u_z}{\partial x^2} + \mu\frac{\partial^2 u_z}{\partial y^2} + (\lambda + \mu)\frac{\partial^2 u_x}{\partial z \partial x} + (\lambda + \mu)\frac{\partial^2 u_y}{\partial z \partial y}$$

which can be written in vector form as:

$$\rho\vec{\boldsymbol{a}} = \vec{\boldsymbol{b}} + \mu\nabla^2\vec{\boldsymbol{u}} + (\lambda + \mu)\nabla(\nabla \cdot \vec{\boldsymbol{u}}), \tag{2.33}$$

and where $\nabla^2\vec{\boldsymbol{u}}$ can also be written as $\nabla(\nabla \cdot \vec{\boldsymbol{u}}) - \nabla \times (\nabla \times \vec{\boldsymbol{u}})$

2.3 Lossless acoustic waves in solids and fluids

Apart from particular cases, it is generally difficult to solve Navier's equations directly. Stokes-Helmholtz decomposition of the displacement field, however, offers an interesting workaround [68, p. 47]. Named $\vec{\boldsymbol{u}}$ the particle displacement associated with an acoustic process and assuming it is twice differentiable in a certain domain, Helmholtz's theorem states that $\vec{\boldsymbol{u}}$ can be decomposed as:

$$\vec{\boldsymbol{u}} = \nabla\phi + \nabla \times \vec{\boldsymbol{\psi}} \quad with \quad \nabla \cdot \vec{\boldsymbol{\psi}} = 0, \tag{2.34}$$

where the first term indicates the gradient of a *scalar potential* and the second term is the rotor of a *vector potential*. In order to uniquely determine $\vec{\boldsymbol{u}}$ from ϕ and $\vec{\boldsymbol{\psi}}$, the additional condition $\nabla \cdot \vec{\boldsymbol{\psi}} = 0$ shall be included [73, p. 275]. Both terms ϕ and $\vec{\boldsymbol{\psi}}$ can be part of the same solution, but it is easier to understand the physical meaning by considering them separately.

The first term is usually referred as the *irrotational term*, since if it exists a potential ϕ so that $\vec{\boldsymbol{u}} = \nabla\phi$ then $\nabla \times \vec{\boldsymbol{u}} = \nabla \times \nabla\phi = 0$. Since $\vec{\boldsymbol{u}}$ describes the displacement of a particle, a null rotor entails that the particle cannot rotate as a rigid body. Substituting $\vec{\boldsymbol{u}} = \nabla\phi$ into (2.33) and assuming the effect of the body force negligible:

$$\nabla^2\vec{\boldsymbol{u}} = \frac{1}{c_L^2}\frac{\partial^2\vec{\boldsymbol{u}}}{\partial t^2}, \tag{2.35}$$

where:

$$c_L = \sqrt{\frac{\lambda + 2\mu}{\rho}}. \tag{2.36}$$

Equation (2.35) is a linear lossless wave equation. Since it has been obtained under the condition $\nabla \times \vec{\boldsymbol{u}} = 0$, it describes waves that cause the medium to strain but not to rotate, and they are called *longitudinal* or *irrotational waves*. An important observation comes from the property of the Laplacian:

$$\nabla\nabla^2\phi = \nabla^2\nabla\phi = \nabla^2\vec{\boldsymbol{u}} = \frac{1}{c_L^2}\frac{\partial^2\vec{\boldsymbol{u}}}{\partial t^2} = \frac{1}{c_L^2}\frac{\partial^2\nabla\phi}{\partial t^2} = \nabla\frac{1}{c_L^2}\frac{\partial^2\phi}{\partial t^2}. \tag{2.37}$$

Therefore, the same equation can be solved equivalently either for the particle displacement or its scalar potential.

The second term is usually referred as the *transverse term* since if $\vec{\boldsymbol{u}} = \nabla \times \vec{\boldsymbol{\psi}}$ then $\nabla \cdot \vec{\boldsymbol{u}} = 0$. Indeed, recalling equation (2.15) and considering that:

$$\nabla \cdot \vec{\boldsymbol{u}} = \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z} = \varepsilon_{xx} + \varepsilon_{yy} + \varepsilon_{yy} = I\varepsilon_1 = 0, \tag{2.38}$$

the field described by the transverse term cannot support variations of the particles' volume. Therefore, there can be normal strain only if it sums to zero with the normal strain along the other directions in space. Using the previous equation, substituting ε_{xx}, ε_{yy}, ε_{zz} into (2.32) and neglecting the effect of the body force it yields:

$$\nabla^2 \vec{\boldsymbol{u}} = \frac{1}{c_T^2} \frac{\partial^2 \vec{\boldsymbol{u}}}{\partial t^2}, \tag{2.39}$$

where:

$$c_T = \sqrt{\frac{\mu}{\rho}}. \tag{2.40}$$

The (2.39) is again a linear lossless wave equation with propagation velocity c_T. Its solutions are *equivoluminal waves* also called *transverse waves*. As for the scalar potential, the property of the Laplacian entails:

$$\nabla \times \nabla^2 \vec{\boldsymbol{\psi}} = \nabla^2 \nabla \times \vec{\boldsymbol{\psi}} = \nabla^2 \vec{\boldsymbol{u}} = \frac{1}{c_T^2} \frac{\partial^2 \vec{\boldsymbol{u}}}{\partial t^2} = \frac{1}{c_T^2} \frac{\partial^2 \nabla \times \vec{\boldsymbol{\psi}}}{\partial t^2} = \nabla \times \frac{1}{c_T^2} \frac{\partial^2 \vec{\boldsymbol{\psi}}}{\partial t^2}. \tag{2.41}$$

Therefore, the same equation can be solved equivalently for the particle displacement or for its vector potential. Transverse and longitudinal waves can coexist at the same time (their displacement simply sum up) but they propagate at different speeds depending on the value of ρ, μ and λ.

To yield equations (2.37) and (2.41), it has been tacitly assumed that the acceleration $\vec{\boldsymbol{a}}$ can be simply written as $\vec{\boldsymbol{a}} = \partial^2 \vec{\boldsymbol{u}} / \partial t^2$. This identity, however, is valid only if the spatial variations of displacement and velocity fields are negligible in respect of the time variations. In case of not negligible variations, a more accurate expression would be $\vec{\boldsymbol{a}} = \partial^2 \vec{\boldsymbol{u}} / \partial t^2 + (\vec{\boldsymbol{v}} \cdot \nabla)\vec{\boldsymbol{v}}$.

When the medium is an ideal inviscid liquid, the shear modulus μ is zero and the (2.33) yields:

$$\rho \frac{\partial^2 \vec{\boldsymbol{u}}}{\partial t^2} = \vec{\boldsymbol{b}} + \lambda \nabla (\nabla \cdot \vec{\boldsymbol{u}}), \tag{2.42}$$

which does not allow transverse waves. Since the shear modulus is null, equation (2.24) returns:

$$\sigma_{xx} = \sigma_{yy} = \sigma_{zz} = \lambda (\varepsilon_{xx} + \varepsilon_{yy} + \varepsilon_{zz}) = -p, \tag{2.43}$$

where p is the *acoustic pressure* assumed opposite to the normal stress by definition. From the previous equation and (2.13), the pressure in an inviscid liquid can be written as:

$$p = -\lambda \nabla \cdot \vec{\boldsymbol{u}}. \tag{2.44}$$

Plugging (2.44) into (2.42) and using the definition of the scalar potential yields:

$$\rho \frac{\partial^2 \nabla \phi}{\partial t^2} = \vec{\boldsymbol{b}} - \nabla p, \tag{2.45}$$

which, for negligible body force, can be rewritten as:

$$p = -\rho \frac{\partial^2 \phi}{\partial t^2}, \tag{2.46}$$

where the acoustic pressure for an inviscid liquid is given as a function of the scalar potential. The particle velocity can be found from the definition of displacement:

$$\vec{\boldsymbol{v}} = \nabla \frac{\partial \phi}{\partial t}. \tag{2.47}$$

Combining the two previous equations, it is possible to relate pressure and particle velocity:

$$-\nabla p = \rho \frac{\partial \vec{\boldsymbol{v}}}{\partial t}. \tag{2.48}$$

Interestingly, from (2.46), (2.47) and (2.37), it is easily shown that both pressure and particle velocity satisfy the same wave equation as for the scalar potential:

$$\nabla^2 p = \frac{1}{c_L} \frac{\partial^2 p}{\partial t^2} \tag{2.49a}$$

$$\nabla^2 \vec{\boldsymbol{v}} = \frac{1}{c_L} \frac{\partial^2 \vec{\boldsymbol{v}}}{\partial t^2}. \tag{2.49b}$$

To summarise, under the condition that:

- the equilibrium density ρ is constant in time,
- the amplitude of the physical quantities associated with the vibroacoustic process (particle displacement, particle velocity, pressure) is small,
- the effect of the body forces is negligible,

it is possible to solve the lossless vibroacoustic problem by solving the same wave equation for a number of different physical quantities. In particular, for an inviscid liquid domain, scalar potential, pressure, particle displacement, and particle velocity can be used. For a solid domain, instead, scalar potential and particle displacement can be used for the irrotational term, while vector potential and particle displacement can be used for the transverse term. For all the cases above, the particular solution is defined only when a sufficient number of initial and boundary conditions are defined.

2.4 Plane waves and wavenumber vector

An *acoustic plane wave* is a wave whose acoustic quantities have constant amplitude and phase on any plane perpendicular to the propagation direction [65, p. 122]. The lossless wave equation introduced in the previous section has the general form:

$$\nabla^2 \xi = \frac{1}{c} \frac{\partial^2 \xi}{\partial t^2}, \tag{2.50}$$

where ξ is some scalar or vector quantity and c is the related propagation velocity. Harmonic planar waves are solutions to (2.50) and offer a simple introduction to a very meaningful quantity that remains fundamental in other non-trivial cases: the *wavenumber vector*. If we assume a plane wave travelling in a certain direction, a solution to (2.50) as a function of the position $P \equiv x, y, z$ has the form:

$$\xi(P,t) = Ae^{i(k_x x + k_y y + k_z z - \omega t)}, \tag{2.51}$$

where A is the complex amplitude and

$$\frac{\omega^2}{c^2} = k_x^2 + k_y^2 + k_z^2. \tag{2.52}$$

For the case above, the *wavenumber vector* is defined as:

$$\vec{\boldsymbol{k}} = k_x\vec{\boldsymbol{x}} + k_y\vec{\boldsymbol{y}} + k_z\vec{\boldsymbol{z}}, \tag{2.53}$$

where $\vec{\boldsymbol{x}}$, $\vec{\boldsymbol{y}}$, and $\vec{\boldsymbol{z}}$ are the unitary versors of the reference system. The position vector can be written as:

$$\vec{\boldsymbol{r}} = x\vec{\boldsymbol{x}} + y\vec{\boldsymbol{y}} + z\vec{\boldsymbol{z}}. \tag{2.54}$$

Therefore, (2.51) is equivalent to:

$$\xi(P,t) = Ae^{i(\vec{\boldsymbol{k}}\cdot\vec{\boldsymbol{r}} - \omega t)}. \tag{2.55}$$

Since, at a given time t, $\vec{\boldsymbol{k}}$ can be written as $\nabla(\vec{\boldsymbol{k}}\cdot\vec{\boldsymbol{r}})$, recalling the meaning of the gradient, $\vec{\boldsymbol{k}}$ represents a vector perpendicular to the planes with constant amplitude and phase. Hence, the wavenumber vector points to the direction of the propagation and $k_x/|\vec{\boldsymbol{k}}|$, $k_y/|\vec{\boldsymbol{k}}|$, $k_z/|\vec{\boldsymbol{k}}|$ are the cosines of the propagation direction with respect to the given reference system. The modulus of the wavenumber vector depends on wave frequency and medium properties and its SI unit is $[m^{-1}]$.

As shown in sections 2.8.3 and 4.1, the wavenumber is fundamental to describing the dispersive behaviour of the waveguides.

Plane waves offer a further simple geometrical representation for waves associated with the scalar potential ϕ and the vector potential $\vec{\boldsymbol{\psi}}$ as defined in (2.34). Considering first the former and choosing the reference system so that the propagation direction lies along x, the potential can be written as:

$$\phi(x,y,z,t) = \Phi e^{i(k_x x - \omega t)}. \tag{2.56}$$

The associated displacement $\nabla\phi$ is:

$$\vec{\boldsymbol{u}}_\phi(x,y,z,t) = U_\phi e^{i(k_x x - \omega t)}\vec{\boldsymbol{x}} \tag{2.57}$$

and, therefore, the scalar potential displacement has components only in the direction of the propagation. Similarly, the wave associated with the vector potential can be written as:

$$\vec{\boldsymbol{\psi}}(x,y,z,t) = \Psi_x e^{i(k_x x - \omega t)}\vec{\boldsymbol{x}} + \Psi_y e^{i(k_x x - \omega t)}\vec{\boldsymbol{y}} + \Psi_z e^{i(k_x x - \omega t)}\vec{\boldsymbol{z}}. \tag{2.58}$$

The associated displacement $\nabla \times \vec{\boldsymbol{\psi}}$ is:

$$\vec{\boldsymbol{u}}_\psi(x,y,z,t) = U_{\psi_y} e^{i(k_x x - \omega t)}\vec{\boldsymbol{y}} + U_{\psi_z} e^{i(k_x x - \omega t)}\vec{\boldsymbol{z}} \tag{2.59}$$

and, therefore, the direction of the displacement is orthogonal to the direction of the propagation.

2.5 Acoustic intensity

The pressure of a harmonic plane wave of amplitude P along x can be written as:

$$p(x,t) = Pe^{i(kx - \omega t)} \tag{2.60}$$

and, from (2.48) and (2.52), the related particle velocity is:

$$v(x,t) = \frac{P}{\rho c} e^{i(kx-\omega t)} = V e^{i(kx-\omega t)}. \tag{2.61}$$

An important quantity often used to characterise acoustic signals is the *acoustic intensity* [135, p. 61], which is the time-averaged work per unit area done on an element of fluid. It is defined as:

$$I = \frac{1}{T} \int_0^T p(t) |\vec{\boldsymbol{v}}(t)| dt. \tag{2.62}$$

For the plane wave given by (2.60) and (2.61) the intensity is:

$$I = \frac{P^2}{2\rho c}, \tag{2.63}$$

where a sign can be added to specify the propagation direction. Usually, the intensity is given using logarithmic scales in decibels. The *intensity level* is defined as:

$$IL = 10 \log (I/I_{ref}). \tag{2.64}$$

where I_{ref} is a reference intensity value.

Section 5.2.2 describes the procedure to acquire the soundbank using a hydrophone. The output quantities of microphones and hydrophones are usually proportional to the input pressure. For this reason, measurements are frequently given as pressure with respect to a reference pressure value P_{ref}. The *sound pressure level* is defined as:

$$SPL = 20 log(P_e/P_{ref}) \tag{2.65}$$

where P_e is the measured *effective* amplitude of the pressure. For the plane wave given by (2.60) and (2.61) $P_e = P/\sqrt{2}$ and $I = P_e^2/\rho c$ so the intensity level

IL and the sound pressure level *SPL* return the same numeric value. Interestingly, the same result is valid also for spherical waves but cannot be assumed true in general [65, p. 127].

2.6 Boundary conditions and acoustic impedance

When a wave reaches the boundary of a medium, its propagation can be altered. This section describes a few possible scenarios assuming simplified boundary configurations. Boundary conditions can be assigned by imposing specific values to some acoustic quantities or applying specific acoustic conditions at the separation surfaces. Acoustic quantities are generally imposed in terms of stress, particle displacement or particle velocity, or a combination of them [73, p. 311]. Two of the simplest cases are the *solid boundary*, where null displacement is assumed, and *pressure release boundary*, where null pressure is assumed. Sometimes, time-varying conditions can be imposed to account for acoustic sources.

When a boundary separates different materials, several scenarios are possible. It is essential to understand which stress, displacement, or velocity component must be assumed continuous according to material and interface properties. In general, continuity of stress means that there is no net force acting on the separation boundary, while continuity of displacement means that the mediums remain in contact along a specific direction[74, pp. 92–97]. The application of these two statements depends on the specific setup. For instance, when the boundary separates a solid from a liquid domain, the continuity of shear stress entails a null value at the interface, while displacement is continuous only in the normal direction. On the other hand, for a bonded solid-solid setup, the boundary shear stress is usually not null, and displacement is continuous also along the tangent directions.

A plane wave that reaches the boundary can be *reflected*, generating new waves that bounce back, or, in case of propagation into a second medium, *refracted*, generating new waves beyond the separation surface with the same frequency but different direction and velocity. Although other options are possible [68, p. 45], they are unnecessary for our analysis.

Material properties play a key role in understanding propagation behaviour. For instance, in an inviscid liquid, a plane wave reaching the boundary generates no more than a reflected wave and, in the case of propagation into a second liquid, no more than one refracted. However, when one or both half-spaces are solid, the simple scenario described above becomes more complicated. As shown in section 2.3, in a solid body, longitudinal and transverse waves can coexist and propagate independently. When a longitudinal or transverse wave encounters a separation boundary, it can experience *mode conversion*, meaning that a longitudinal wave can generate a transverse wave and vice-versa. In other words, the separation boundary introduces a coupling between transverse and longitudinal waves. As one can imagine, the analytical complexity increases with the number of separation boundaries and the complexity of the boundary shapes. In some cases, the waves generated have peculiar propagation characteristics, and they are specifically named. *Rayleigh waves* [73, p. 325], generated at the edge of a half-space, or *Love waves* [73, p. 380], generated in a thin solid layer laying on top of a solid half-space, are two examples.

For this research, two cases are significant: the first one is a solid half-space with pressure release boundary conditions, and the second one is the analysis of a liquid-solid interface. The liquid is assumed inviscid, and the solid is supposed to be isotropic.

2.6.1 Plane wave in a solid half-space with pressure release boundary

This section focuses on a plane wave that propagates through a solid half-space with pressure release boundaries [73, p. 312]. Without loss of generality, it is possible to assume that the boundary matches with the plane $y = 0$ (the *horizontal plane*[2]) and that the wavenumber vector lies on the plane xy (the *vertical plane*[2]). Thus, longitudinal motion develops completely on the vertical plane, while the shear motion has components in both the vertical plane and its orthogonal direction. Given the reference system chosen, no wave quantity depends on z. Boundary conditions can be written as:

[2] Terminology is borrowed from seismology

$$\sigma_{yy}(x,0,z,t) = 0 \tag{2.66a}$$

$$\sigma_{yx}(x,0,z,t) = 0 \tag{2.66b}$$

$$\sigma_{yz}(x,0,z,t) = 0. \tag{2.66c}$$

From the Helmholtz decomposition (2.34) and since $\partial/\partial z = 0$, the particle displacement components can be written as:

$$u_x = \frac{\partial \phi}{\partial x} + \frac{\partial \psi_z}{\partial y} \qquad u_y = \frac{\partial \phi}{\partial y} - \frac{\partial \psi_z}{\partial x} \qquad u_z = \frac{\partial \psi_y}{\partial x} - \frac{\partial \psi_x}{\partial y}. \tag{2.67}$$

Combining previous equations with (2.27) yields:

$$\begin{aligned}
\sigma_{xx} &= \lambda\left(\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2}\right) + 2\mu\left(\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\psi_z}{\partial x\partial y}\right) & \sigma_{xy} &= \mu\left(2\frac{\partial^2\phi}{\partial x\partial y} + \frac{\partial^2\psi_z}{\partial y^2} - \frac{\partial^2\psi_z}{\partial x^2}\right) \\
\sigma_{yy} &= \lambda\left(\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2}\right) + 2\mu\left(\frac{\partial^2\phi}{\partial y^2} - \frac{\partial^2\psi_z}{\partial x\partial y}\right) & \sigma_{yz} &= \mu\left(\frac{\partial^2\psi_y}{\partial x\partial y} - \frac{\partial^2\psi_x}{\partial y^2}\right) \\
\sigma_{zz} &= \lambda\left(\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2}\right) & \sigma_{zx} &= \mu\left(\frac{\partial^2\psi_y}{\partial x^2} - \frac{\partial^2\psi_x}{\partial x\partial y}\right).
\end{aligned} \tag{2.68}$$

Equations (2.66), (2.67) and (2.68) describe a problem of wave motion that can be decoupled into two independent parts: a *plane strain* and a *SH-wave* motion [73, p. 314]. Indeed, the displacement components u_x and u_y and the stress components σ_{yy} and σ_{yx} depend only on ϕ and ψ_z; ϕ and ψ_z, in turn, are regulated by the wave equation for scalar potential (2.37) and by the z component of the wave equation for the vector potential (2.41). Similarly, the displacement component u_z and the stress component σ_{yz} depend only on ψ_x and ψ_y; ψ_x and ψ_y are regulated by the x and y components of the wave equation for the vector potential (2.41). It follows that plain strain and SH-wave can be investigated separately.

For what concerns the plain strain, solutions that include both incident and reflected waves of the potentials ϕ and ψ_z for a harmonic plane wave can be written as:

$$\phi = \phi_i + \phi_r = \Phi_i e^{ik_P(x\sin\theta_P - y\cos\theta_P - c_P t)} + \Phi_r e^{ik_P(x\sin\theta_P + ycos\theta_P - c_P t)} \tag{2.69}$$

$$\psi_z = \psi_{zi} + \psi_{zr} = \Psi_{zi} e^{ik_S(x\sin\theta_S - y\cos\theta_S - c_S t)} + \Psi_{zr} e^{ik_S(x\sin\theta_S + ycos\theta_S - c_S t)}, \tag{2.70}$$

where:

$$k_P^2 = \frac{\omega^2}{c_P^2} = \frac{\rho\omega^2}{\lambda + 2\mu} \qquad k_S^2 = \frac{\omega^2}{c_S^2} = \frac{\rho\omega^2}{\mu}. \tag{2.71}$$

Recalling the meaning of the wavenumber vector given in 2.4, the direction cosines are used to express the direction of the plane waves. The angles θ_P and θ_S indicate the incidence and reflection angles of the waves for longitudinal and transverse components, respectively. The amplitude of the incident scalar potential is indicated with Φ_i, and the amplitude of the reflected scalar potential is indicated with Φ_r. The same notations are adopted for ψ_z. Applying the boundary conditions (2.66a) (2.66b) and dropping the term $e^{-i\omega t}$ yields:

$$\sigma_{yy}(x,0,z,t) = 0: \tag{2.72}$$

$$k_P^2\left(2\sin^2\theta_P - \frac{\lambda+2\mu}{\mu}\right)(\Phi_i + \Phi_r)e^{ik_P x\sin\theta_P} - k_S^2\sin 2\theta_S(\Psi_{zi} - \Psi_{zr})e^{ik_S x\sin\theta_S} = 0$$

$$\sigma_{xy}(x,0,z,t) = 0: \tag{2.73}$$

$$k_P^2\sin 2\theta_P(\Phi_i - \Phi_r)e^{ik_P x\sin\theta_P} - k_S^2\cos 2\theta_S(\Psi_{zi} + \Psi_{zr})e^{ik_S x\sin\theta_S} = 0.$$

The previous conditions are true for any value of x, hence the following condition, which is the *Snell's law* for elastic waves, applies:

$$k_P\sin\theta_P = k_S\sin\theta_S \quad or \quad \frac{\sin\theta_P}{\sin\theta_S} = \frac{c_P}{c_S} = \sqrt{\frac{\lambda+2\mu}{\mu}} = \kappa \tag{2.74}$$

Separating the reflection terms from the others, equations (2.72) and (2.73) can be organised in the following form:

$$\begin{bmatrix}\Phi_r \\ \Psi_{zr}\end{bmatrix} = \begin{bmatrix}S_{PP} & S_{SP} \\ S_{PS} & S_{SS}\end{bmatrix}\begin{bmatrix}\Phi_i \\ \Psi_{zi}\end{bmatrix} \tag{2.75}$$

where the matrix of the terms S_{ij} denotes the *scattering matrix*. Its coefficients can be written as:

$$S_{PP} = \frac{sin2\theta_P \sin 2\theta_S - \kappa^2 \cos^2 2\theta_S}{\sin 2\theta_P \sin 2\theta_S + \kappa^2 \cos^2 2\theta_S} \tag{2.76a}$$

$$S_{SP} = \frac{-2\kappa^2 \sin 2\theta_S \cos 2\theta_S}{\sin 2\theta_P \sin 2\theta_S + \kappa^2 \cos^2 2\theta_S} \tag{2.76b}$$

$$S_{PS} = \frac{2 \sin 2\theta_P \cos 2\theta_S}{\sin 2\theta_P \sin 2\theta_S + \kappa^2 \cos^2 2\theta_S} \tag{2.76c}$$

$$S_{SS} = \frac{\sin 2\theta_P \sin 2\theta_S - \kappa^2 \cos^2 2\theta_S}{\sin 2\theta_P \sin 2\theta_S + \kappa^2 \cos^2 2\theta_S} \tag{2.76d}$$

The scattering matrix and Snell's law establish a relationship between the incident and reflected potentials on the vertical plane. In particular, when a vertical longitudinal wave (*P-wave*) hits the boundary, it can be reflected both as a P longitudinal wave and as a vertical transverse wave (*SV-wave*). The direction of the reflected waves is given by Snell's law, while the amplitude by the scattering matrix. A similar argument is valid for incident SV-waves.

A few cases are particularly interesting to analyse (figure 2.5). For example, assuming a pure perpendicular incident P-wave ($\theta_P = 0$), only S_{PP} and S_{PS} must be considered. The value of the former is -1, while the latter is null. Hence, the reflected wave has a reverse phase and opposite direction, while no SV-wave is reflected. Besides, if $\sin 2\theta_P \sin 2\theta_S = \kappa^2 \cos^2 2\theta_S$ (it can happen for up to two values of θ_P depending on the Poisson ratio), no P-wave is reflected while an SV-wave is reflected at an angle $\theta_S < \theta_P$. For other values of θ_P, both P and SV-waves are reflected from a single P-wave. When the incident wave is an SV-wave, the angle of the reflected P-wave θ_P is wider than the incidence angle θ_S. In order to have θ_P in the admissible range $[0, \pi/2]$, θ_S should fall in the range $[0, \arcsin 1/\kappa]$. The *critical angle* is $\theta_S = \arcsin 1/\kappa$ [73, p. 321]. To understand what happens beyond the critical angle, the reflected P component can be rewritten as:

$$\phi_r = \Phi_r e^{ik_P(\alpha x + \beta y)} e^{-i\omega t}, \tag{2.77}$$

and from (2.74):

$$\beta^2 = 1 - \alpha^2 = 1 - \sin^2\theta_P = 1 - \frac{k_T^2}{k_P^2}\sin^2\theta_T. \tag{2.78}$$

For angles beyond the critical angle, $\beta = \pm i\xi$ and, taking the only solution with physical meaning, the reflected P-wave can be written as:

$$\phi_r = \Phi_r e^{-\xi y} e^{i(k_L \alpha x - \omega t)}. \tag{2.79}$$

Hence, the P-wave propagates in the x direction and attenuates exponentially along y, a behaviour similar to the *evanescent waves* in waveguides (sections 2.8.3 and 4.1).

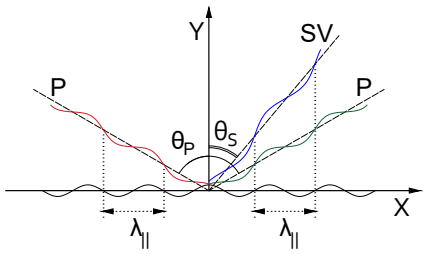


Figure 2.5: *Reflection from the pressure release boundary of a solid half-space. Boundary conditions: null stress at the boundary. A P-wave can be reflected as a P-wave and as an SV-wave. The wavenumber vector has a continuous parallel component* $1/\lambda_{||}$ *at the boundary.*

Reflections of SH-waves can be analysed with similar steps. However, it can be proved [73, p.316] that SH-waves simply reflect themselves with no mode conversion or amplitude variation. Hence, if a shear wave with an arbitrary polarisation strikes the boundary of a simple half-space, the SH component is simply reflected with the same incident angle and the same amplitude. In contrast, the SV component sees a change in its amplitude and part of its energy is converted into a reflected P-wave.

It is remarked that when a wave impinges a separation boundary, both incident and reflected wavefronts share the same acoustic quantities at the boundary and, in particular, this observation is valid for the parallel component of the wavenumber

vector $1/\lambda_{\|} = k_P \sin\theta_P = k_S \sin\theta_T$. Snell's law, thus, quantifies the discontinuity of the orthogonal component of the wavenumber vector at the boundary. The simple reflection of a plane wave at the boundary of a solid half-space and the related P-SV conversion is shown in figure 2.5. In the next section, it is shown how waves are also refracted in a second medium.

2.6.2 Plane wave through a liquid-solid interface

When a boundary surface separates two mediums, an incident wave generates both reflected and refracted waves. This section describes the case of an inviscid liquid half-space separated from a solid half-space by a flat boundary and a plane wave propagating from the liquid to the solid domain.

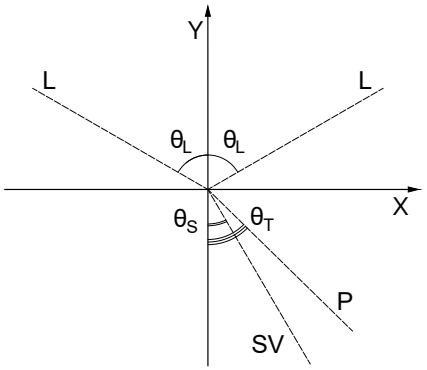


Figure 2.6: *Propagation of a plane P-wave from a liquid half-space to a solid half-space. Boundary conditions: null shear stress, continuity of normal stress and normal displacement. The incident P-wave is refracted as a P-wave and an SV-wave.*

The reference system is again chosen with wavenumber vectors on the xy plane and the boundary on $y = 0$ (figure 2.6). The subscripts (l) and (s) differentiate between the liquid and solid domains where necessary. The analysis is similar to the one in the previous section, but some energy is transferred through the boundary in the form of refracted waves. It is remarked that the incident and the reflected waves in the liquid domain can only be of type P. Since the shear stress in the liquid domain is null, boundary conditions translate into null shear stress, continuity of normal stress and continuity of normal displacement:

$$\sigma_{(l)yy}(x,0,z,t) = \sigma_{(s)yy}(x,0,z,t) \tag{2.80a}$$

$$\sigma_{(s)yx}(x,0,z,t) = 0 \tag{2.80b}$$

$$\sigma_{(s)yz}(x,0,z,t) = 0 \tag{2.80c}$$

$$u_{(l)y}(x,0,z,t) = u_{(s)y}(x,0,z,t). \tag{2.80d}$$

As for the pressure release boundary, the problem can be decomposed into two parts: a displacement in the vertical xy plane and a shear displacement in the z direction. From equations (2.67), (2.68) and (2.80c), it follows that the shear displacement in the z direction is null. The sum of the incident potential γ_i and the reflected potential γ_r in the liquid domain is:

$$\gamma = \gamma_i + \gamma_r = \Gamma_i e^{ik_L(x\sin\theta_L - y\cos\theta_L - c_L t)} + \Gamma_r e^{ik_L(x\sin\theta_L + ycos\theta_L - c_L t)}, \tag{2.81}$$

while for the solid domain the potentials are:

$$\phi = \Phi e^{ik_P(x\sin\theta_P - y\cos\theta_P - c_P t)} \tag{2.82}$$

$$\psi_z = \Psi_z e^{ik_S(x\sin\theta_S - y\cos\theta_S - c_S t)}. \tag{2.83}$$

The related boundary conditions can be written as:

$$\sigma_{(l)yy}(x,0,z,t) = \sigma_{(s)yy}(x,0,z,t) : \tag{2.84a}$$
$$\lambda_{(l)}k_L^2(\Gamma_i + \Gamma_r)e^{ik_L x\sin\theta_L} =$$
$$(\lambda_{(s)} + 2\mu\cos^2\theta_P)\Phi k_P^2 e^{ik_P x\sin\theta_P} + 2\mu\Psi_z k_S^2 \sin\theta_S\cos\theta_S e^{ik_S x\sin\theta_S}$$

$$\sigma_{(s)yx}(x,0,z,t) = 0 : \tag{2.84b}$$
$$2\Phi k_P^2 \sin\theta_P\cos\theta_P e^{ik_P x\sin\theta_P} + (1 - 2\cos^2\theta_S)\Psi_z k_S^2 e^{ik_S x\sin\theta_S} = 0$$

$$u_{(l)y}(x,0,z,t) = u_{(s)y}(x,0,z,t) : \tag{2.84c}$$
$$i(-\Gamma_i + \Gamma_r)k_L\cos\theta_L e^{ik_L x\sin\theta_L} =$$
$$-i\Phi k_P\cos\theta_P e^{ik_P x\sin\theta_P} + i\Psi_z k_S\sin\theta_S e^{ik_S x\sin\theta_S}.$$

Snell's law is found again imposing the previous equations valid for any x:

$$k_L \sin\theta_L = k_P sin\theta_P = k_S \sin\theta_S. \tag{2.85}$$

The three potential component amplitudes Γ_r, Φ, Ψ_z can be found from the incident potential Γ_i:

$$\begin{bmatrix} \Gamma_r \\ \Phi \\ \Psi_z \end{bmatrix} = \begin{bmatrix} S_L \\ S_P \\ S_S \end{bmatrix} \Gamma_i \tag{2.86}$$

where S_L, S_P and S_S can be found from equations (2.84) [68, p. 92].

2.6.3 Specific acoustic impedance

A meaningful acoustic quantity often used to characterise different materials is the *specific acoustic impedance* [65, p. 126, 286]. For simplicity, only the case of propagation in an inviscid liquid is considered. The specific acoustic impedance z is defined as the ratio of the acoustic pressure to the value of the particle velocity along its vector direction:

$$z = p/v. \tag{2.87}$$

It is a measure of the opposition of the medium to wave propagation. Its unit is [Pa s/m], sometimes called *rayl*. In the general case, the specific acoustic impedance is a complex number (real "acoustic resistance" and imaginary "acoustic reactance"), but unless otherwise specified, its value is assumed to be the real one given for plane waves. Assuming a plane pressure wave P and a reference system so that P propagates along x, the pressure can be written as:

$$p(x,t) = Pe^{i(kx-\omega t)}, \tag{2.88}$$

and from (2.48) and (2.52), the related particle velocity is:

$$v(x,t) = \frac{P}{\rho c} e^{i(kx-\omega t)}. \tag{2.89}$$

Hence, for a plane wave, the characteristic acoustic impedance is the real quantity:

$$z = \rho c. \tag{2.90}$$

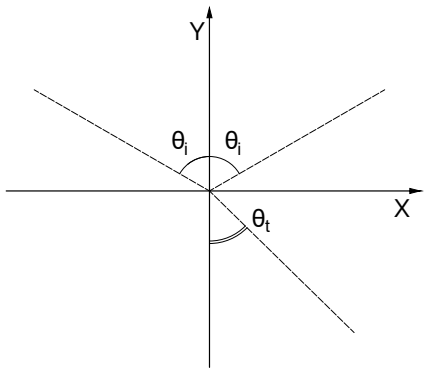


Figure 2.7: *Liquid-liquid boundary. Boundary conditions: continuity of normal stress and continuity of normal displacement.*

The concept of specific acoustic impedance is useful to analyse the transfer of energy at the boundary. Considering two fluid half-spaces with a flat separation surface (figure 2.7), the boundary conditions can be imposed as continuity of pressure (or normal stress) and continuity of normal velocity (equivalent to normal displacement). From (2.46) and (2.47) pressure and particle velocity can be written as a function of the scalar potential as:

$$p = -\rho \frac{\partial^2 \gamma}{\partial t^2} \qquad \vec{\boldsymbol{v}} = \nabla \frac{\partial \gamma}{\partial t}. \tag{2.91}$$

Potentials in the two liquid half-spaces l_1 and l_2 are:

$$\gamma_{(l_1)} = \Gamma_i e^{ik_{(l_1)}(x\sin\theta_i - y\cos\theta_i - c_{(l_1)}t)} + \Gamma_r e^{ik_{(l_1)}(x\sin\theta_i + y\cos\theta_i - c_{(l_1)}t)} \tag{2.92}$$

$$\gamma_{(l_2)} = \Gamma_t e^{ik_{(l_2)}(x\sin\theta_t - y\cos\theta_t - c_{(l_2)}t)} \tag{2.93}$$

where the subscripts (l_1), (l_2) indicate the mediums and the subscripts i, r and t are associated with the incident, the reflected and the transmitted wave, respectively. Repeating similar steps as in the previous sections, continuity of pressure and normal velocity can be written as:

$$p_{(l_1)}(x,0,z,t) = p_{(l_2)}(x,0,z,t): \tag{2.94a}$$
$$\rho_{(l_1)}(\Gamma_i + \Gamma_r) = \rho_{(l_2)}\Gamma_t$$
$$v_{y(l_1)}(x,0,z,t) = v_{y(l_2)}(x,0,z,t): \tag{2.94b}$$
$$\frac{1}{c_{(l_1)}}(\Gamma_i - \Gamma_r)\cos\theta_i = \frac{1}{c_{(l_2)}}\Gamma_t \cos\theta_t$$

where ρ indicates the density and c the sound speeds. The ratio of the reflected pressure amplitude P_r to the incident pressure amplitude P_i is called *reflection coefficient* R, while the ratio of transmitted pressure amplitude P_t to the incident pressure amplitude P_i is called *transmission coefficient* T. From the boundary conditions:

$$R + T = 1, \tag{2.95}$$

which, together with (2.94), yields:

$$R = \frac{P_r}{P_i} = \frac{\Gamma_r}{\Gamma_i} = \frac{z_2/z_1 - cos\theta_t/\cos\theta_i}{z_2/z_1 + \cos\theta_t/\cos\theta_i} \qquad T = \frac{P_t}{P_i} = 1 - R, \tag{2.96}$$

where from Snell's law:

$$\cos\theta_t = \sqrt{1 - \frac{c^2_{(l_2)}}{c^2_{(l_1)}}\sin^2\theta_i}. \tag{2.97}$$

Hence, the reflection and transmission coefficients depend only on the incidence angle and on the specific acoustic impedance z_1 and z_2. Along with R and T, sometimes the intensity reflection and transmission coefficients R_I and T_I are given. Recalling identity (2.63), for a plane wave:

$$R_I = \frac{I_r}{I_i} = |R|^2 \qquad T_I = \frac{I_t}{I_i} = \frac{z_1}{z_2}|T|^2. \tag{2.98}$$

When the incidence angle is orthogonal to the surface and the two specific acoustic impedances are similar, the reflection is minimum and most of the intensity of the wave is transmitted through the surface. On the contrary, when the mismatch between z_1 and z_2 increases, the intensity of the reflected wave increases and the intensity of the transmitted wave decreases.

2.7 Phase velocity and group velocity

This section introduces two important concepts used to analyse reverberation phenomena in waveguides: *phase velocity* and *group velocity*. Let ξ be a certain physical quantity propagating in the x direction:

$$\xi(x,t) = \xi(x-ct). \tag{2.99}$$

A value of ξ at point x_1 and time t_1 can be found at point x_2 and time t_2 if:

$$x_1 - ct_1 = x_2 - ct_2. \tag{2.100}$$

Hence, a specific value of ξ moves a distance $x_2 - x_1$ along x in a time $t_2 - t_1$. The quantity:

$$c_p = c = \frac{x_2 - x_1}{t_2 - t_1} \tag{2.101}$$

is called *phase velocity* and describes the velocity at which a point of the wave with a constant phase moves along the propagation direction. For harmonic waves:

$$\xi(x,t) = \Xi e^{i(kx-\omega t)}, \tag{2.102}$$

and the phase velocity can be written as:

$$c_p = \omega/k, \tag{2.103}$$

where ω is the harmonic frequency and k is the wavenumber. When the modulus of the wavenumber is linearly proportional to ω, the phase velocity remains constant in frequency, and a medium is said to be *non-dispersive*. A signal whose spectrum spreads across a certain frequency range sees its components travel at the same speed, and no distortion is experienced. On the other hand, if k is a non-linear function of the frequency, the ratio of ω to k does not remain constant, and components of the same signal at different frequencies propagate with different velocities. In this case, a medium is called *dispersive*. Analysis of the phase velocity in dispersive mediums can be misleading. Indeed, waves can exhibit phase velocity far beyond or below the speed at which their energy is actually propagated (figure 2.8).

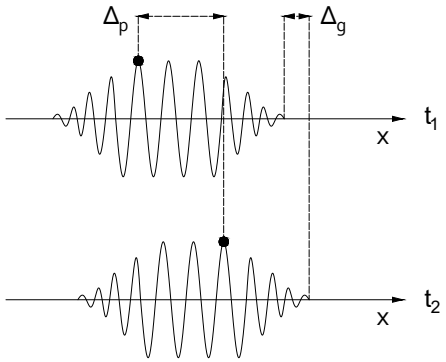


Figure 2.8: *Phase and group velocity in a dispersive medium. Δ_p indicates the temporal shift associated with the phase velocity, while Δ_g indicates the one associated with the group velocity.*

A more meaningful quantity is the *group velocity* [73, p. 59]. To introduce this concept, let $\xi(t)$ be a narrow band pulse with central frequency $\overline{\omega}$. Its Fourier transform $\hat{\xi}(\omega)$ is defined as[3]:

[3] For the acoustic model developed in chapters 2 and 4, the Fourier transform is defined using the basis $e^{i\omega t}/\sqrt{2\pi}$, which is the prevalent definition adopted in our acoustics references. In the other chapters, the basis is defined as $e^{-i\omega t}/\sqrt{2\pi}$. As long as the anti-transform is defined accordingly, a different definition makes no difference.

$$\hat{\xi}(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} \xi(t) e^{i\omega t} dt. \tag{2.104}$$

and the associated propagating pulse can be written as:

$$\xi(t,x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} \hat{\xi}(\omega) e^{i(k(\omega)x - \omega t)} d\omega. \tag{2.105}$$

If the band of the pulse is narrow enough, it is possible to assume that the range of the wavenumber is also narrow. Hence, $k(\omega)$ can be expanded in Taylor's series truncated at the first derivative term:

$$k(\omega) \simeq k(\overline{\omega}) + \frac{dk(\omega)}{d\omega}\Big|_{\overline{\omega}} (\omega - \overline{\omega}). \tag{2.106}$$

Therefore, the propagating pulse in the time domain can be written as:

$$\xi(t,x) = \frac{e^{i\left(k(\overline{\omega}) - \frac{\partial k(\omega)}{\partial \omega}\Big|_{\overline{\omega}} \overline{\omega}\right)x}}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} \hat{\xi}(\omega) e^{i\left(\frac{\partial k(\omega)}{\partial \omega}\Big|_{\overline{\omega}} \omega x - \omega t\right)} d\omega. \tag{2.107}$$

where it is shown that, in the "propagation factor", all the frequency components of the signal $\hat{\xi}(\omega)$ roughly move along x with the same velocity

$$v_g = \frac{\partial \omega}{\partial k}, \tag{2.108}$$

where the derivative is calculated for $\omega = \overline{\omega}$. v_g is called *group velocity*, and a visual comparison against phase velocity is reported in figure 2.8. In a dispersive medium, the phase shift Δ_p in an inteval $\Delta t = t_2 - t_1$ does not match the energy shift Δ_g given by the group velocity. An identical definition holds when a wideband pulse is considered [65, p. 257],[73, p. 65]. However, since the frequency components spread across a wider frequency range, the propagation introduces a higher degree of distortion. Thus, an averaged group velocity value can be found by considering those components that carry most of the signal energy.

2.8 Analysis of circular straight rigid lossless waveguides

When the acoustic energy is confined in an enclosure, standing waves can be stimulated, and their characteristics depend on the shape and materials of the enclosure, on the initial conditions and on the boundary conditions [65, p.246]. If no energy is dissipated, the standing waves sustain the oscillations without any energy source. The modes associated with the standing waves determine the acoustic behaviour of the enclosure, and the values of the acoustic quantities can be determined as the superimposition of different modes. The number of modes stimulated and their amplitudes depend on the initial conditions [73, p. 37],[136].

If the enclosure is partially open, standing waves combine with propagation along some specific directions, and the enclosure behaves as a waveguide [135, p. 75]. Since propagation transfers acoustic energy, an acoustic source is necessary to sustain a continuous signal even when the domain is assumed to be lossless. The number and the amplitude of the stimulated modes depend on the initial conditions and the acoustic sources; their dispersive behaviour depends on the geometries, the materials, and the boundary conditions. Propagation in partially open enclosures can be simulated using finite element software [122, p. 321]. However, as seen in section 2.1, full FE methods are generally computationally expensive, and the results might be difficult to analyse and integrate. Large geometries, for instance, are generally simplified in subsections to reduce the calculations required [137],[116]. From this point of view, although analytical solutions suffer important limitations, they offer a clearer explanation of the underlying acoustic phenomena and are certainly faster to calculate and easier to integrate. This section provides the analysis of a simple case study as an introduction to acoustic reverberation and for comparison with the model developed in chapter 4.

2.8.1 Modelling propagation

The geometry analysed in this section is illustrated in figure 2.9. It consists of an infinite inviscid fluid cylinder with a radius W. Clearly, it is convenient to adopt a cylindrical reference system with the z axis matching the cylinder axis. The

boundary is assumed perfectly rigid and, as seen in section 2.6, this assumption entails that the normal component of displacement is null for $r = W$:

$$u_r|_W = 0. \tag{2.109}$$

Using (2.48), the previous equation becomes:

$$\left.\frac{\partial p}{\partial r}\right|_W = 0. \tag{2.110}$$

Solutions of equation (2.49a) can be a family of functions that are twice differentiable with respect to r,θ,z,t, but, for modal analysis, harmonic solutions are considered:

$$p(r,\theta,z,t)\Big|_{\bar{\omega}} = \hat{p}(r,\theta,z)\Big|_{\bar{\omega}} e^{-i\bar{\omega}t} \tag{2.111}$$

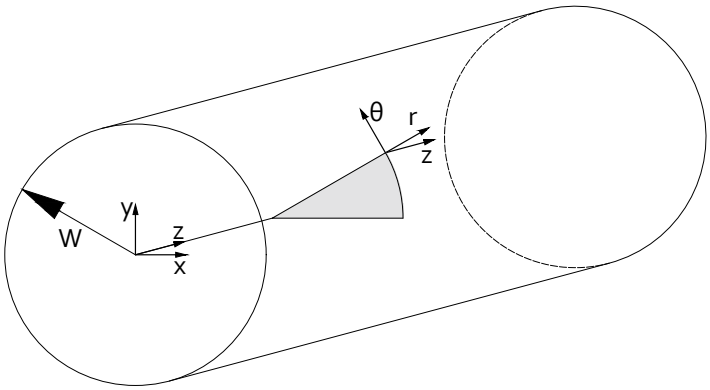


Figure 2.9: *Straight lossless circular waveguide with rigid boundaries*

It is worth noting that if p_1 and p_2 are solutions of (2.49a), their linear combination is still a solution, and this is valid for any number of solutions at different frequencies. Hence, for a continuous spectrum:

$$p(r,\theta,z,t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} \hat{p}(r,\theta,z,\omega) e^{-i\omega t} d\omega, \tag{2.112}$$

where $\hat{p}(r,\theta,z,\omega)$ is the Fourier transform of $p(r,\theta,z,t)$. Substitution of (2.112) into (2.49a) yields:

$$\nabla^2 \int_{-\infty}^{+\infty} \hat{p}(r,\theta,z,\omega) e^{-i\omega t} d\omega = \frac{1}{c^2} \frac{\partial^2}{\partial t^2} \int_{-\infty}^{+\infty} \hat{p}(r,\theta,z,\omega) e^{-i\omega t} d\omega, \tag{2.113}$$

which is equivalent to:

$$\int_{-\infty}^{+\infty} \left(\nabla^2 \hat{p}(r,\theta,z,\omega) + k^2 \hat{p}(r,\theta,z,\omega)\right) e^{-i\omega t} d\omega = 0, \tag{2.114}$$

with the *wavenumber* k defined as $k = \omega/c$. The integral in (2.114) is certainly null when null is its integrand. Thus, if a function $\hat{p}$ satisfies the *Helmholtz equation*:

$$\nabla^2 \hat{p} + k^2 \hat{p} = 0 \tag{2.115}$$

for any frequency of interest, it can also be assumed solution for the (2.114). In cylindrical coordinates, equation (2.115) can be written as:

$$\frac{\partial^2 \hat{p}}{\partial r^2} + \frac{1}{r}\frac{\partial \hat{p}}{\partial r} + \frac{1}{r^2}\frac{\partial^2 \hat{p}}{\partial \theta^2} + \frac{\partial^2 \hat{p}}{\partial z^2} + k^2 \hat{p} = 0. \tag{2.116}$$

The most common method to solve (2.116) is the spatial *separation of variables* [65, p. 247],[102, p. 121]. The solution $\hat{p}$ is assumed to be the product of three functions, each of them dependent only on one of the three space variables:

$$\hat{p}(r,\theta,z) = R(r)\Theta(\theta)Z(z), \tag{2.117}$$

where in the previous equation ω has been omitted. Plugging (2.117) into (2.116) and dividing by $R(r)\Theta(\theta)Z(z)$ yields:

$$\frac{\partial^2 Z(z)}{\partial z^2} = -k_z^2 Z(z), \tag{2.118}$$

and

$$\frac{1}{R(r)}\frac{\partial^2 R(r)}{\partial r^2} + \frac{1}{rR(r)}\frac{\partial R(r)}{\partial r} + \frac{1}{r^2\Theta(\theta)}\frac{\partial^2 \Theta(\theta)}{\partial \theta^2} = -q^2. \tag{2.119}$$

with $k^2 = q^2 + k_z^2$ and either k_z or q unknown. k_z represents the magnitude of the wavenumber vector along the propagation direction while q is its orthogonal component. Equation (2.118) regulates the propagation along the z axis and a harmonic propagative solution is:

$$Z(z) = \bar{Z}_+ e^{ik_z z} + \bar{Z}_- e^{-ik_z z}, \tag{2.120}$$

where only the first term should be considered for propagation along the positive direction of z. Since no boundary condition is specified along z, the value of k_z is determined by its relation with k, which depends on the driving frequency and on the boundary conditions along the other directions. Equation (2.119) can be further decomposed by multiplying for r^2 and separating terms in r and θ. That is:

$$\frac{\partial^2 \Theta(\theta)}{\partial \theta^2} = -n^2 \Theta(\theta), \tag{2.121}$$

and

$$r^2\frac{\partial^2 R(r)}{\partial r^2} + r\frac{\partial R(r)}{\partial r} + (q^2 r^2 - n^2)R(r) = 0. \tag{2.122}$$

Solutions of equation (2.121) describe the variation of the pressure as a function of the angle θ and are of the form:

$$\Theta(\theta) = \bar{\Theta} e^{in\theta}, \tag{2.123}$$

with θ in an arbitrary interval $[\bar{\theta}, \bar{\theta}+2\pi]$. Given the physical meaning, pressure in $\bar{\theta}$ should be equal to the pressure in $\bar{\theta}+2\pi$, and this particular kind of boundary condition requires that the separation constant n must be an integer. Equation (2.122) has the form of a Bessel equation [102, p. 466] of order n. With n integer, solutions of (2.122) are of the form:

$$R(r) = \bar{R}J_n(qr) + \bar{\bar{R}}Y_n(qr), \tag{2.124}$$

where $J_n(r)$ and $Y_n(r)$ indicate the Bessel function of the first kind and of the second kind of order n, respectively. Since $Y_n(r)$ has a singularity in $r=0$ (it would mean infinite pressure in $r=0$), $\bar{\bar{R}}$ must be assumed zero and the general solution for $R(r)$ is:

$$R(r) = \bar{R}J_n(qr). \tag{2.125}$$

To find the solution, the values of q must be determined, and its value, together with k, also provides the values of k_z, which regulates the propagation along z. q clearly depends on the value of n as both are part of the equation (2.122). The integer n can be used as an index to identify all the q that satisfy (2.122) for a certain n. With n fixed, the set of allowable values of q can be determined from (2.122) using the Sturm-Liouville theory [138],[102, p. 363, 374]. Equation (2.122) can be turned in Sturm-Liouville form:

$$-(g(r)p_r')' + q(r)p_r = \lambda w(r)p_r \tag{2.126}$$

dividing by r and using the following assumptions:

$$g(r) = r, \qquad q(r) = \frac{n^2}{r}, \qquad w(r) = r, \qquad \lambda = q^2, \tag{2.127}$$

with r defined in $[0, W]$ and Newmann boundary conditions expressed by (2.110). Since the Sturm-Liouville theory can be applied to the (2.122), it is possible to identify an infinite number of discrete nonnegative real values $\lambda_0 < \lambda_1 < \ldots < \lambda_m < \ldots$

(the *eigenvalues*), each related to a particular solution $R(r)$ (the *eigenvectors*). Thus, from (2.126), per each index n, it is possible to extract a set of values of q that can be sorted in ascending order and identified by an index m. In the following, a particular value of q is indicated as q_{nm}, the corresponding eigenfunction is indicated as $R_{nm}(r)$, and the related value of k_z as $k_{z_{nm}}$. The actual values of the eigenvalues λ_m can be found from the boundary condition:

$$\frac{\partial R_{nm}(W)}{\partial r} = \bar{R}_{nm} J_n'(q_{nm} W) = 0, \tag{2.128}$$

which means finding the zeros of the first derivative of the Bessel function of order n. For n integer, J_n can be written in the integral form as [139, pp. 5.5.1–5.5.5]:

$$J_n(r) = \frac{1}{\pi} \int_0^{\pi} cos(rcos\phi - n\phi) d\phi, \tag{2.129}$$

and, to find the first derivative, it is possible to use the following identity:

$$\frac{\partial J_n(r)}{\partial r} = \frac{n}{r} J_n(r) - J_{n+1}(r). \tag{2.130}$$

The zeros for negative values of n can be found from the zeros for positive values of n using the following formula:

$$J_{-n}(r) = (-1)^n J_n(r). \tag{2.131}$$

The only special case is for $n = 0$. Indeed, $J_0'(0) = 0$ means $q_{00} = 0$ and $J_0(q_{00} r) = 1$ in the whole interval $[0, W]$.

From (2.120), (2.123) and (2.125), equation (2.117) for a single modal solution in the frequency domain can be written as:

$$\hat{p}_{nm}(r, \theta, z, \omega) = M_{nm} e^{in\theta} J_n(q_{nm} r) e^{ik_{z_{nm}} z}, \tag{2.132}$$

where $\bar{R}_{nm}$, $\bar{\theta}_n$, and $\bar{Z}_+$ have been merged in the complex term M_{nm} (with $n \in \mathbb{Z}$ $[...,-2,-1,0,1,2,...]$, $m \in \mathbb{N}$ $[1,2,...]$) and ω dependence is included in q_{nm}, $k_{z_{nm}}$ and, in general, also in M_{nm}.

2.8.2 Modelling a cross-sectional pressure source

A cross-sectional pressure source placed in $z = 0$ can be modelled by assuming its pressure distribution as a special boundary condition on the circular cross-section. The amplitude of the modes is then chosen in order to match the pressure source given. Considering (2.132) in $z = 0$, each mode can be written as:

$$\hat{p}_{0_{nm}}(r,\theta) = M_{nm} e^{in\theta} J_n(q_{nm} r), \tag{2.133}$$

where M_{nm} are the modal amplitudes to be determined according to a certain source. In the 2D domain of the section $z = 0$ ($r \in [0,W]$, $\theta \in [0,2\pi]$), the set of functions given by (2.133) with rigid or pressure release boundary conditions form a complete and orthogonal *basis* in respect of the following definition of *inner product*:

$$\langle \zeta_1(r,\theta), \zeta_2(r,\theta) \rangle = \int_0^{2\pi} \int_0^W \zeta_1(r,\theta) \zeta_2^*(r,\theta) r \, dr d\theta, \tag{2.134}$$

where $*$ indicates the complex conjugate [138]. The *orthogonality* statement means that the basis functions:

$$\zeta_{nm} = e^{in\theta} J_n(q_{nm} r) \qquad n \in \mathbb{Z} \quad m \in \mathbb{N} \tag{2.135}$$

satisfy the condition:

$$\langle \zeta_{nm}(r,\theta), \zeta_{n'm'}(r,\theta) \rangle = \int_0^{2\pi} \int_0^W \zeta_{nm}(r,\theta) \zeta_{n'm'}^*(r,\theta) r \, dr \, d\theta = A_{nn'mm'} \delta_{nn'} \delta_{mm'}, \tag{2.136}$$

with δ_{nm} being the *Kronecker delta* and $A_{nn'mm'}$ a constant. Indeed:

$$\langle \zeta_{nm}(r,\theta), \zeta_{n'm'}(r,\theta) \rangle = \int_0^{2\pi} e^{i(n-n')\theta} d\theta \int_0^W J_n(q_{nm}r) J_{n'}(q_{n'm'}r) r dr, \quad (2.137)$$

where the integral in $d\theta$ is null for any $n' \neq n$. Thus, assuming $n = n'$, to solve the second integral we can recall that if $B(r)$ is solution of the Bessel equation (2.122) then:

$$r q_{nm}^2 B_n(q_{nm}r) = -\frac{\partial}{\partial r}\left\{ r \frac{\partial B_n(q_{nm}r)}{\partial r} \right\} + \frac{n^2}{r} B_n(q_{nm}r), \quad (2.138)$$

and

$$\begin{aligned}
& q_{nm}^2 \int_a^b r B_n(q_{nm}r) B_n(q_{nm'}r) dr = \\
& \int_a^b \left[-\frac{\partial}{\partial r}\left\{ r \frac{\partial B_n(q_{nm}r)}{\partial r} \right\} + \frac{n^2}{r} B_n(q_{nm}r) \right] B_n(q_{nm'}r) dr = \\
& -\int_a^b \frac{\partial}{\partial r}\left\{ r \frac{\partial B_n(q_{nm}r)}{\partial r} \right\} B_n(q_{nm'}r) dr + \int_a^b \frac{n^2}{r} B_n(q_{nm}r) B_n(q_{nm'}r) dr = \\
& -\left[r \frac{\partial B_n(q_{nm}r)}{\partial r} B_n(q_{nm'}r) \right]_a^b + \int_a^b r \frac{\partial B_n(q_{nm}r)}{\partial r} \frac{\partial B_n(q_{nm'}r)}{\partial r} dr \\
& + \int_a^b \frac{n^2}{r} B_n(q_{nm}r) B_n(q_{nm'}r) dr = \\
& -\left[r \frac{\partial B_n(q_{nm}r)}{\partial r} B_n(q_{nm'}r) \right]_a^b + \left[r B_n(q_{nm}r) \frac{\partial B_n(q_{nm'}r)}{\partial r} \right]_a^b \\
& -\int_a^b \frac{\partial}{\partial r}\left\{ r \frac{\partial B_n(q_{nm'}r)}{\partial r} \right\} B_n(q_{nm}r) dr + \int_a^b \frac{n^2}{r} B_n(q_{nm}r) B_n(q_{nm'}r) dr = \\
& -\left[r \left(\frac{\partial B_n(q_{nm}r)}{\partial r} B_n(q_{nm'}r) - \frac{\partial B_n(q_{nm'}r)}{\partial r} B_n(q_{nm}r) \right) \right]_a^b \\
& + \int_a^b \left[-\frac{\partial}{\partial r}\left\{ r \frac{\partial B_n(q_{nm'}r)}{\partial r} \right\} + \frac{n^2}{r} B_n(q_{nm'}r) \right] B_n(q_{nm}r) dr = \\
& -\left[r \left(\frac{\partial B_n(q_{nm}r)}{\partial r} B_n(q_{nm'}r) - \frac{\partial B_n(q_{nm'}r)}{\partial r} B_n(q_{nm}r) \right) \right]_a^b \\
& q_{nm'}^2 \int_a^b r B_n(q_{nm}r) B_n(q_{nm'}r) dr.
\end{aligned} \quad (2.139)$$

Therefore:

$$(q_{nm}^2 - q_{nm'}^2)\int_a^b rB_n(q_{nm}r)B_n(q_{nm'}r)dr = \left[r\left(\frac{\partial B_n(q_{nm'}r)}{\partial r}B_n(q_{nm}r) - \frac{\partial B_n(q_{nm}r)}{\partial r}B_n(q_{nm'}r)\right)\right]_a^b. \tag{2.140}$$

Applying the above equation to the current case where $B_n = J_n$, $a = 0$, and $b = W$, it follows that the inner product (2.137) for $n = n'$ and $m \neq m'$ can be assumed always null if:

$$J_n(q_{nm}W) = 0 \quad or \quad \frac{\partial J_n(q_{nm}W)}{\partial r} = 0, \tag{2.141}$$

where the first is a pressure release and the second is a rigid boundary condition. Hence, since the latter of the (2.141) has been assumed true, the functions (2.135) are mutually orthogonal. The *completeness* of the basis comes from the fact that the eigenfunctions associated with regular boundary conditions on bounded domains automatically form a complete basis [102, p. 379]. The result of the scalar product is generally not unitary, but it can be normalised by the normalisation factors:

$$N_{00} = \frac{1}{W\sqrt{\pi}} \qquad n = m = 0 \tag{2.142a}$$

$$N_{nm} = \frac{q_{nm}/\sqrt{\pi}}{\sqrt{(q_{nm}W)^2 - n^2}\,J_n(q_{nm}W)} \qquad otherwise. \tag{2.142b}$$

and, with these assumptions, the basis is said to be *orthonormal*.

The coefficients M_{nm} can be determined to match the specific pressure spatial distribution of the source. Since the functions in (2.135) form a complete and orthonormal basis, a source $S(r,\theta,t)$ with an arbitrary pressure distribution placed in $z = 0$ and whose Fourier transform is:

$$\hat{S}(r,\theta,\omega) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} S(r,\theta,t)e^{i\omega t}dt \tag{2.143}$$

can be decomposed as a linear combination of the function of the basis. The coefficients of the linear combination are the magnitudes M_{nm} and, generally, they are frequency dependent. They are determined from:

$$M_{nm}(\omega) = \frac{\int_0^{2\pi}\int_0^W \hat{S}(r,\theta,\omega)\zeta_{nm}^*(r,\theta)r\,dr d\theta}{\int_0^{2\pi}\int_0^W \zeta_{nm}(r,\theta)\zeta_{nm}^*(r,\theta)r\,dr d\theta}, \tag{2.144}$$

where the formula above is a direct consequence of the orthogonality of the basis. Note that if the pressure space distribution does not depend on the frequency, the value of the coefficient M_{nm} can be assumed constant over the whole frequency range.

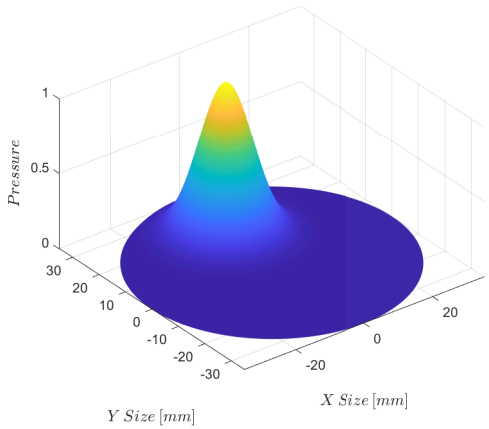


Figure 2.10: *Example of normalised spatial pressure distribution placed in* $z = 0$.

An example of the spatial distribution of a normalised pressure source placed in $z = 0$ is shown in figure 2.10. Figures 2.11 represent the amplitudes of two of the related modes calculated with (2.144). Once the coefficients have been determined, the pressure source can be expanded as:

$$\hat{S}(r,\theta,\omega) = \sum_{n=-\infty}^{+\infty}\sum_{m=1}^{+\infty} M_{nm}(\omega)\zeta_{nm}(r,\theta). \tag{2.145}$$

Figures 2.12 show how the sum of a sufficiently high number of modes approximates the same shape of the pressure source given. The pressure values along the waveguide can be found from:

$$\hat{p}(r,\theta,z,\omega) = \sum_{n=-\infty}^{+\infty} \sum_{m=1}^{+\infty} M_{nm}(\omega)\zeta_{nm}(r,\theta)e^{ik_{z_{nm}}z}. \tag{2.146}$$

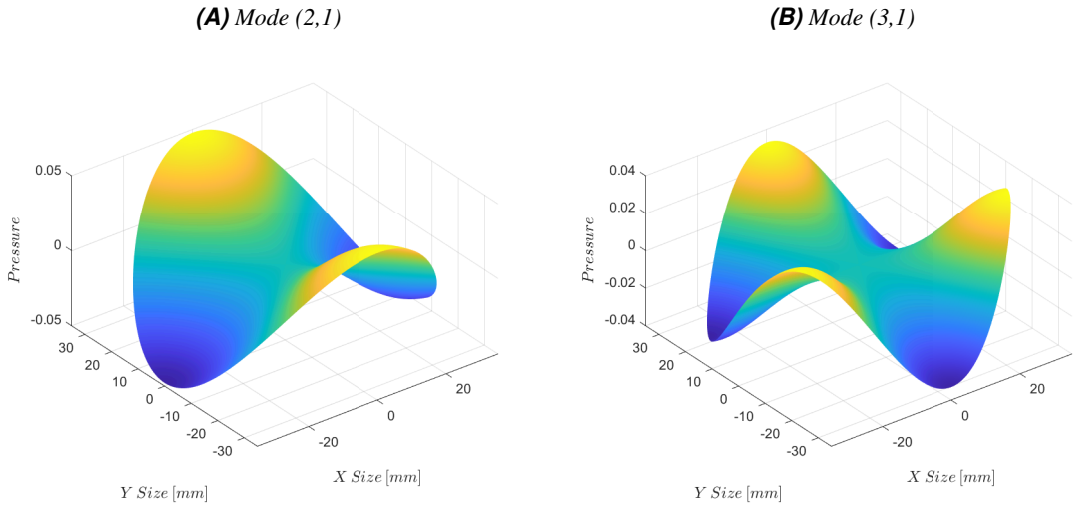


Figure 2.11: *Source modal components calculated for the cross section $z = 0$ and for the pressure source reported in figure 2.10. A) Mode: $n = 2$, $m = 1$. B) Mode: $n = 3$, $m = 1$.*

Pressure in the time domain can be obtained from (2.112), which yields:

$$p(r,\theta,z,t) = \frac{1}{\sqrt{2\pi}} \sum_{n=-\infty}^{+\infty} \sum_{m=1}^{+\infty} \int_{-\infty}^{+\infty} M_{nm}(\omega)\zeta_{nm}(r,\theta)e^{ik_{z_{nm}}z}e^{-i\omega t}d\omega. \tag{2.147}$$

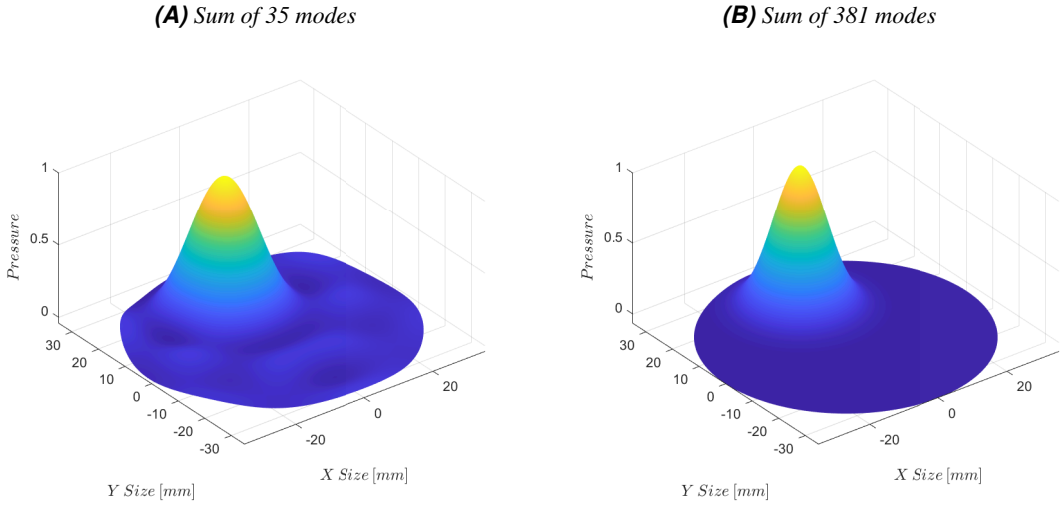


Figure 2.12: *Reconstruction of the pressure source of figure 2.10 using modal decomposition. A) Source approximation using 35 modes. B) Source approximation using 381 modes.*

2.8.3 Dispersive effects

To understand how a waveguide affects the acoustic signal as it propagates along the channel, it is convenient to analyse first a single-frequency harmonic signal. As seen in section 2.8.1, a single harmonic wave at frequency $\overline{\omega}$ can be written as:

$$p_{nm}(r,\theta,z,t)|_{\overline{\omega}} = M_{nm} e^{in\theta} J_n(q_{nm} r) e^{i(k_{z_{nm}} z - \overline{\omega} t)}. \tag{2.148}$$

It has been shown that, given geometries and boundary conditions, values of q_{nm} are fixed and frequency independent per each mode. However, since:

$$\frac{\omega^2}{c^2} = k_{z_{nm}}^2 + q_{nm}^2, \tag{2.149}$$

$k_{z_{nm}}$ is clearly frequency dependent, and the waveguide shows dispersive behaviour in the z direction. From the previous identity, it is possible to distinguish two different scenarios per each mode. The frequency:

$$\Omega_{nm} = c q_{nm} \tag{2.150}$$

is called *cut-on angular frequency* for the mode nm, and it is the transition frequency between two different behaviours of the mode nm. For frequencies above the cut-on, $k_{z_{nm}}$ is real and (2.148) describes a wave propagating along the positive z direction. For frequencies below the cut-on, $k_{z_{nm}}$ is purely imaginary and it can be written as:

$$k_{z_{mn}} = \pm i \sqrt{q_{mn}^2 - \left(\frac{\omega}{c}\right)^2}. \tag{2.151}$$

For the only sign with a physical meaning, it entails:

$$p_{nm}(r,\theta,z,t)|_{\overline{\omega}} = M_{nm} e^{in\theta} J_n(q_{nm} r) e^{-z\sqrt{q_{mn}^2 - \frac{\overline{\omega}^2}{c^2}}} e^{-i(\overline{\omega} t)}, \tag{2.152}$$

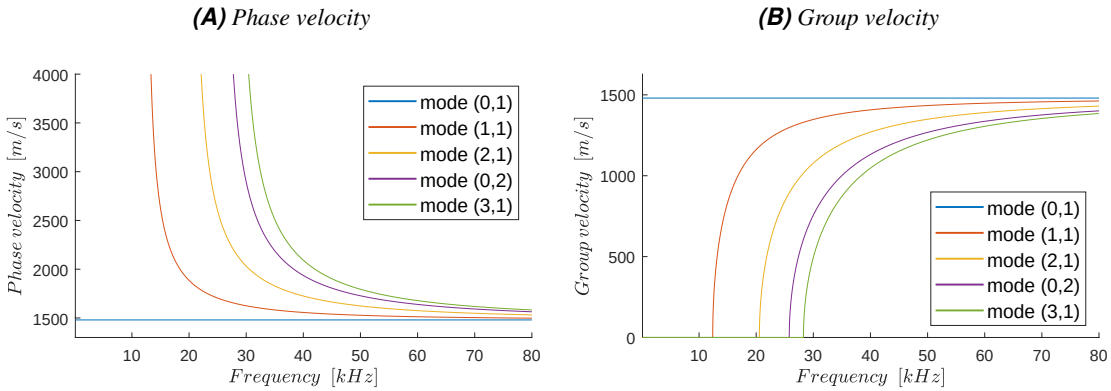


Figure 2.13: *Velocities for an inviscid water cylinder with rigid boundary. $W = 35mm$, $c = 1480m/s$. All the modes (except $(0,0)$) exhibit dispersive behaviour. A) Phase velocity, $v_p \geq c$. B) Group velocity, $v_g \leq c$.*

which describes an *evanescent* standing wave that attenuates exponentially along z and does not propagate energy along the waveguide. Phase velocity and group velocity can be calculated directly from (2.103) and (2.108).

The phase and the group velocities for an inviscid water cylinder, $W = 35mm$, $c = 1480m/s$, are reported in figure 2.13. Modes are represented above the cut-on and, with the exception of the mode $(0,0)$, all the others exhibit dispersive behaviour with variable phase and group velocity. Note how the phase velocity can be higher than the speed of sound in water c and how, around the cut-on, the dispersive behaviour is more pronounced. The wavenumbers k_z as a function of frequency are shown in figure 2.14.

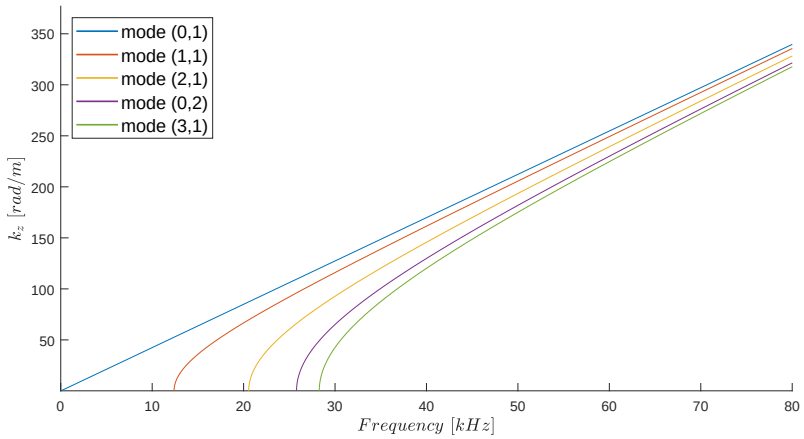


Figure 2.14: *k_z for an inviscid water cylinder with rigid boundary. W = 35 mm, c=1480m/s. All the modes, with the exception of the mode $(0,0)$, are dispersive.*

An interesting physical interpretation is given in [65, p. 249], where (2.148) is approximated as:

$$p_{nm}(r,\theta,z,t)|_{\overline{\omega}} \simeq \frac{M_{nm}}{\sqrt{q_{mn}r}} e^{i(n\theta)} e^{i(\pm q_{mn}r + k_{z_{mn}}z - \overline{\omega}t)} \tag{2.153}$$

using an asymptotic approximation of the Bessel function. Equation (2.153) can be interpreted as a decomposition of the mode in two conic waves whose directions with respect to the z axis are given by:

$$\pm \arctan\left(\frac{q_{nm}}{k_{z_{nm}}}\right) = \pm \arctan\left(\frac{q_{nm}}{\sqrt{q_{nm}^2 - \frac{\omega^2}{c^2}}}\right) = \pm \arctan\left(\frac{1}{\sqrt{1 - \frac{\omega^2}{\Omega^2}}}\right) \tag{2.154}$$

Hence, when the frequency ω is close to the cut-on, the directions of the conic waves tend to be orthogonal to z, explaining the reason for the low group velocity and the high phase velocity along z. On the contrary, when the frequency ω is much higher than the cut-on, the directions of the conic waves tend to z, and both phase and group velocities tend to c.

2.9 Effect of elastic boundaries

In the previous section, assuming rigid walls yielded simple boundary conditions, but this hypothesis might not be reasonable in a real context. As seen in section 2.6.3, a rigid boundary condition is a good approximation only when the mismatch of the specific acoustic impedances is high (e.g. metal/air). When this condition is not satisfied (e.g. metal/water), the walls cannot be assumed rigid since a certain amount of energy is transferred through the interface. The propagation in different mediums, however, emerges as a unique process influenced by all the materials.

The literature reports several applications for waveguides with elastic boundaries, such as pipe inspections [140], in-pipe communications [66] and noise reduction [141]. Applications are often supported by numeric simulations, but the analytic approach is useful to have a deeper understanding of the underlying physics [128]. The mathematical formulation provided in this section is a fundamental block of

the model developed in chapter 4, where several additional aspects are developed and analysed.

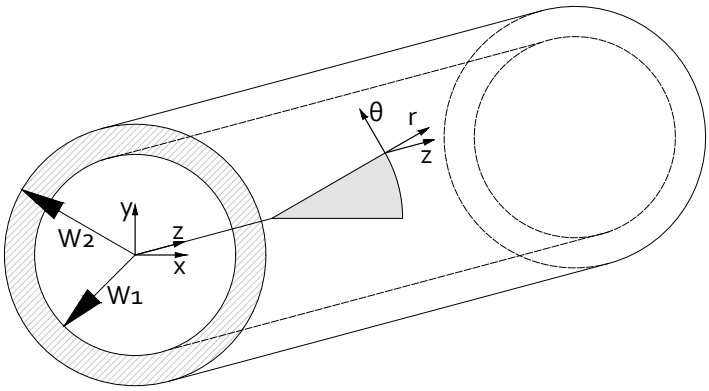


Figure 2.15: *Circular elastic waveguide*

2.9.1 Equations of a straight lossless circular waveguide with elastic boundaries

This section follows an approach first developed by Del Grosso [142] for axisymmetric waveguides and then further developed by Gazis [143] for the general case. A straight, circular waveguide with a lossless solid elastic shell filled with some inviscid lossless fluid is considered. The outer surface of the shell is assumed to be surrounded by air at ambient pressure. The inner and the outer radii are W_1 and W_2, respectively (Fig. 2.15).

In order to determine the acoustic quantities, it is useful to set the equations for the potentials. As seen in section 2.3, the shell admits a displacement vector with both vector and scalar potentials, while the displacement for the internal fluid is obtained from the scalar potential only. Adopting a system of cylindrical coordinates with the z axis matching the axis of the waveguide, and repeating the same steps seen in section 2.8.1 to yield the Helmholtz equation (2.115), equations (2.37) and (2.41) can be written as:

$$\frac{\partial^2\hat{\phi}}{\partial r^2}+\frac{1}{r}\frac{\partial\hat{\phi}}{\partial r}+\frac{1}{r^2}\frac{\partial^2\hat{\phi}}{\partial\theta^2}+\frac{\partial^2\hat{\phi}}{\partial z^2}+k_\phi^2\hat{\phi}=0 \tag{2.155a}$$

$$\frac{\partial^2\hat{\gamma}}{\partial r^2}+\frac{1}{r}\frac{\partial\hat{\gamma}}{\partial r}+\frac{1}{r^2}\frac{\partial^2\hat{\gamma}}{\partial\theta^2}+\frac{\partial^2\hat{\gamma}}{\partial z^2}+k_\gamma^2\hat{\gamma}=0 \tag{2.155b}$$

$$\frac{\partial^2\hat{\vec{\boldsymbol{\psi}}}}{\partial r^2}+\frac{1}{r}\frac{\partial\hat{\vec{\boldsymbol{\psi}}}}{\partial r}+\frac{1}{r^2}\frac{\partial^2\hat{\vec{\boldsymbol{\psi}}}}{\partial\theta^2}+\frac{\partial^2\hat{\vec{\boldsymbol{\psi}}}}{\partial z^2}+k_\psi^2\hat{\vec{\boldsymbol{\psi}}}=0, \tag{2.155c}$$

where ϕ indicates the scalar potential for the liquid, while γ and $\vec{\boldsymbol{\psi}}$ are the scalar and the vector potentials for the solid, respectively. Equation (2.155c) is a vector equation and, since in polar coordinates the unit vectors $\vec{\boldsymbol{r}}$ and $\vec{\boldsymbol{\theta}}$ are not independent, it can be rewritten as:

$$\frac{\partial^2\hat{\psi}_r}{\partial r^2}+\frac{1}{r}\frac{\partial\hat{\psi}_r}{\partial r}-\frac{\hat{\psi}_r}{r^2}+\frac{1}{r^2}\frac{\partial^2\hat{\psi}_r}{\partial\theta^2}-\frac{2}{r^2}\frac{\partial\hat{\psi}_{s\theta}}{\partial\theta}+\frac{\partial^2\hat{\psi}_r}{\partial z^2}+k_\psi^2\hat{\psi}_r=0 \tag{2.156a}$$

$$\frac{\partial^2\hat{\psi}_\theta}{\partial r^2}+\frac{1}{r}\frac{\partial\hat{\psi}_\theta}{\partial r}-\frac{\hat{\psi}_\theta}{r^2}+\frac{1}{r^2}\frac{\partial^2\hat{\psi}_\theta}{\partial\theta^2}+\frac{2}{r^2}\frac{\partial\hat{\psi}_r}{\partial\theta}+\frac{\partial^2\hat{\psi}_\theta}{\partial z^2}+k_\psi^2\hat{\psi}_\theta=0 \tag{2.156b}$$

$$\frac{\partial^2\hat{\psi}_z}{\partial r^2}+\frac{1}{r}\frac{\partial\hat{\psi}_z}{\partial r}+\frac{1}{r^2}\frac{\partial^2\hat{\psi}_z}{\partial\theta^2}+\frac{\partial^2\hat{\psi}_z}{\partial z^2}+k_\psi^2\hat{\psi}_z=0, \tag{2.156c}$$

where the derivative identities $\partial\vec{\boldsymbol{r}}/\partial\theta=\vec{\boldsymbol{\theta}}$ and $\partial\vec{\boldsymbol{\theta}}/\partial\theta=-\vec{\boldsymbol{r}}$ have been used. Equations (2.155) and (2.156) can be solved by separating the variables as for section 2.8.1. However, a few more observations must be pointed out to satisfy the boundary conditions and to solve the coupled equations for $\hat{\psi}_r$ and $\hat{\psi}_\theta$. The form of the solution for all the potential components is assumed to be:

$$\xi(r,\theta,z)=R(r)\Theta(\theta)Z(z). \tag{2.157}$$

Given the geometrical shape, the terms $Z(z)$ describe the propagation along z of the potentials and, considering only the positive direction, it can be assumed of the form:

$$Z(z)=\bar{Z}e^{ik_z z}. \tag{2.158}$$

As seen in section 2.6, Snell's law entails that k_z, which is the component of the

wavenumber vector parallel to the boundary, must be assumed equal for all the potentials. The components $\Theta(\theta)$ provide the angular variation of the potentials, and, given the geometry, they must be $2\pi/n$ periodic. Again, they can be assumed of the form:

$$\Theta(\theta) = \bar{\Theta} e^{in\theta}, \tag{2.159}$$

with $n \in \mathbb{Z}[..., -2, -1, 0, 1, 2, ...]$. However, the continuity of the normal displacement u_r across the liquid-solid boundary imposes a further condition between the $\Theta(\theta)$ components of the potentials. If $\Theta_\phi(\theta) = \bar{\Theta}_\phi e^{in\theta}$, the boundary condition:

$$\frac{\partial \hat{\phi}}{\partial r} = u_{lz} = u_{sz} = \frac{\partial \hat{\gamma}}{\partial r} + \frac{1}{r}\frac{\partial \hat{\psi}_z}{\partial \theta} - \frac{\partial \hat{\psi}_\theta}{\partial z} \qquad r = W_1, \forall\theta, \forall z \tag{2.160}$$

entails:

$$\Theta_\phi(\theta) = \bar{\Theta}_\phi e^{in\theta} \tag{2.161a}$$

$$\Theta_\gamma(\theta) = \bar{\Theta}_\gamma e^{in\theta} \tag{2.161b}$$

$$\Theta_{\psi_r}(\theta) = i\bar{\Theta}_{\psi_r} e^{in\theta} \tag{2.161c}$$

$$\Theta_{\psi_\theta}(\theta) = \bar{\Theta}_{\psi_\theta} e^{in\theta} \tag{2.161d}$$

$$\Theta_{\psi_z}(\theta) = i\bar{\Theta}_{\psi_z} e^{in\theta}, \tag{2.161e}$$

where the phase mismatch has been explicitly shown. Furthermore, $\Theta_{\psi_r}(\theta)$ and $\Theta_{\psi_\theta}(\theta)$ are linked by the coupled equations (2.156a) and (2.156b).

$R_\phi(r)$, $R_\gamma(r)$, $R_{\psi_z}(r)$ can be found as described in section 2.8.1. Except for the liquid potentials, both Bessel functions of the first and second kind should be considered since $r = 0$ is not part of the solid domain. Finally, for $R_{\psi_r}(r)$ and $R_{\psi_\theta}(r)$ a few more steps are required [73, p. 466],[68, p. 198]. Collecting together all the amplitude factors and from the relationships found above, we can assume:

$$\hat{\psi}_r = R_{\psi_r}(r) i e^{in\theta} e^{ik_z z} \tag{2.162a}$$

$$\hat{\psi}_\theta = R_{\psi_\theta}(r) e^{in\theta} e^{ik_z z}. \tag{2.162b}$$

Plugging (2.162) into (2.156a) and (2.156b) yields:

$$r^2\frac{\partial^2 R_{\psi_r}}{\partial r^2}+r\frac{\partial R_{\psi_r}}{\partial r}-R_{\psi_r}-n^2R_{\psi_r}-2nR_{\psi_\theta}+r^2q_\psi^2R_{\psi_r}=0 \tag{2.163a}$$

$$r^2\frac{\partial^2 R_{\psi_\theta}}{\partial r^2}+r\frac{\partial R_{\psi_\theta}}{\partial r}-R_{\psi_\theta}-n^2R_{\psi_\theta}-2nR_{\psi_r}+r^2q_\psi^2R_{\psi_\theta}=0, \tag{2.163b}$$

where $q_\psi^2=k_\psi^2-k_z^2$. Adding and subtracting member to member the two previous equations yields:

$$r^2\frac{\partial^2 \dot{R}(r)}{\partial r^2}+r\frac{\partial \dot{R}(r)}{\partial r}+(q_\psi^2r^2-\dot{n}^2)\dot{R}(r)=0, \tag{2.164}$$

where:

$$\dot{R}=\begin{cases} R_+(r)=R_{\psi_r}(r)+R_{\psi_\theta}(r) \\ R_-(r)=R_{\psi_r}(r)-R_{\psi_\theta}(r) \end{cases} \qquad \dot{n}=\begin{cases} n_+=n+1 \\ n_-=n-1 \end{cases}. \tag{2.165}$$

Equation (2.164) has, once again, the form of a Bessel equation. Therefore:

$$R_+(r)=A_+J_{n-1}(q_\psi r)+B_+Y_{n-1}(q_\psi r) \tag{2.166a}$$

$$R_-(r)=A_-J_{n+1}(q_\psi r)+B_-Y_{n+1}(q_\psi r). \tag{2.166b}$$

The property of *gauge invariance* [68, p. 199] states that any equivoluminal displacement field corresponding to any of R_+, R_- and $R_{\psi z}$ can be obtained by a combination of the other two potentials. Hence, the elimination of one of the above potentials yields an equivalent solution without any loss of generality. Assuming $R_-=0$, the previous equations can be turned into:

$$R_{\psi_r}(r)=CJ_{n+1}(q_\psi r)+DY_{n+1}(q_\psi r) \tag{2.167}$$

$$R_{\psi_\theta}(r)=CJ_{n+1}(q_\psi r)+DY_{n+1}(q_\psi r), \tag{2.168}$$

where $C = A_+/2$ and $D = B_+/2$. Merging everything together and considering only real values of k_z, the solutions for the potentials can be summarised as follows:

$$\hat{\phi} = \Phi J_n(q_\phi r) e^{in\theta} e^{ik_z z} \tag{2.169a}$$

$$\hat{\gamma} = \left[\Gamma_1 J_n(q_\gamma r) + \Gamma_2 Y_n(q_\gamma r)\right] e^{in\theta} e^{ik_z z} \tag{2.169b}$$

$$\hat{\psi}_r = \left[\Psi_1 J_{n+1}(q_\psi r) + \Psi_2 Y_{n+1}(q_\psi r)\right] i e^{in\theta} e^{ik_z z} \tag{2.169c}$$

$$\hat{\psi}_\theta = \left[\Psi_1 J_{n+1}(q_\psi r) + \Psi_2 Y_{n+1}(q_\psi r)\right] e^{in\theta} e^{ik_z z} \tag{2.169d}$$

$$\hat{\psi}_z = \left[\Psi_3 J_n(q_\psi r) + \Psi_4 Y_n(q_\psi r)\right] i e^{in\theta} e^{ik_z z}, \tag{2.169e}$$

where the coefficients are unknown and should be determined from the rest of the boundary and the initial conditions. To obtain the displacement vectors, the following identities for the Bessel functions are applied [102, p.471]:

$$\xi_n'(x) = -\xi_{n+1}(x) + \frac{n}{x}\xi_n(x) \tag{2.170}$$

$$\xi_{n+1}'(x) = \xi_n(x) - \frac{n+1}{x}\xi_{n+1}(x). \tag{2.171}$$

Furthermore, to simplify the tedious calculations, the following compact notations for the Bessel functions are adopted:

$$\overline{A}_n^\alpha = A J_n(q_\alpha r) \tag{2.172a}$$

$$\overline{\overline{AB}}_n^\alpha = A J_n(q_\alpha r) + B Y_n(q_\alpha r). \tag{2.172b}$$

Hence, using (2.170) and (2.171):

$$\frac{\partial A J_n(q_\alpha r)}{\partial r} = -q_\alpha \overline{A}_{n+1}^\alpha + \frac{n}{r}\overline{A}_n^\alpha \tag{2.173a}$$

$$\frac{\partial \left[A J_n(q_\alpha r) + B Y_n(q_\alpha r)\right]}{\partial r} = -q_\alpha \overline{\overline{AB}}_{n+1}^\alpha + \frac{n}{r}\overline{\overline{AB}}_n^\alpha \tag{2.173b}$$

$$\frac{\partial A J_{n+1}(q_\alpha r)}{\partial r} = q_\alpha \overline{A}_n^\alpha - \frac{n+1}{r}\overline{A}_{n+1}^\alpha \tag{2.173c}$$

$$\frac{\partial \left[A J_{n+1}(q_\alpha r) + B Y_{n+1}(q_\alpha r)\right]}{\partial r} = q_\alpha \overline{\overline{AB}}_n^\alpha - \frac{n+1}{r}\overline{\overline{AB}}_{n+1}^\alpha. \tag{2.173d}$$

Plugging the potentials components (2.169) into equation (2.34), the displacement vectors for the liquid domain $\vec{\boldsymbol{u}}_l$ and for the solid domain $\vec{\boldsymbol{u}}_s$ can be obtained. The notations above are used to shorten the analytic expressions:

$$\hat{\vec{\boldsymbol{u}}}_l = e^{ik_z z} e^{in\theta} \begin{bmatrix} \vec{\boldsymbol{r}} & \vec{\boldsymbol{\theta}} & \vec{\boldsymbol{z}} \end{bmatrix} \begin{bmatrix} -q_\phi \overline{\Phi}^{\phi}_{n+1} + \frac{n}{r} \overline{\Phi}^{\phi}_{n} \\ i\frac{n}{r} \overline{\Phi}^{\phi}_{n} \\ ik_z \overline{\Phi}^{\phi}_{n} \end{bmatrix} \tag{2.174}$$

$$\hat{\vec{\boldsymbol{u}}}_s = e^{ik_z z} e^{in\theta} \begin{bmatrix} \vec{\boldsymbol{r}} & \vec{\boldsymbol{\theta}} & \vec{\boldsymbol{z}} \end{bmatrix} \begin{bmatrix} -q_\gamma \overline{\overline{\Gamma_1 \Gamma_2}}^{\gamma}_{n+1} + \frac{n}{r} \overline{\overline{\Gamma_1 \Gamma_2}}^{\gamma}_{n} - \frac{n}{r} \overline{\overline{\Psi_3 \Psi_4}}^{\psi}_{n} - ik_z \overline{\overline{\Psi_1 \Psi_2}}^{\psi}_{n+1} \\ i\frac{n}{r} \overline{\overline{\Gamma_1 \Gamma_2}}^{\gamma}_{n} - k_z \overline{\overline{\Psi_1 \Psi_2}}^{\psi}_{n+1} - i\frac{n}{r} \overline{\overline{\Psi_3 \Psi_4}}^{\psi}_{n} + iq_\psi \overline{\overline{\Psi_3 \Psi_4}}^{\psi}_{n+1} \\ ik_z \overline{\overline{\Gamma_1 \Gamma_2}}^{\gamma}_{n} + q_\psi \overline{\overline{\Psi_1 \Psi_2}}^{\psi}_{n} \end{bmatrix} \tag{2.175}$$

2.9.2 Boundary conditions and wavenumbers

The analytic expressions for the displacement vectors determined in the previous section contain seven unknown constants, namely Φ, Γ_1, Γ_2, Ψ_1, Ψ_2, Ψ_3, Ψ_4. Per each mode, to find the actual value of the modal displacement, the value of the constants must be determined according to the boundary and the initial conditions. As already pointed out, the total displacement is the summation of the displacement of every single mode. Finding the constants, however, depends on the actual value of k_z (which in turn depends on n). In other words, for a given frequency ω and index n, there exists only a set of (infinite) m discrete values of k_z for which the integration constants can be determined. Per each frequency, each value of k_z represents the z component of the wavenumber vector of the mode (n,m). Hence, determining the values of k_z per each frequency yields the dispersion curves of the waveguide and, consequently, the characterisation of the propagation behaviour. Differently from the rigid boundary case, however, it

seems not possible to find an analytic expression for k_z, and its value should be determined through a numeric approach.

As from section 2.6, for the geometry reported in (Fig. 2.15) the boundary conditions can be summarised as:

- continuity of normal displacement at solid-liquid interface ($r = W_1$, 1 equation);
- continuity of normal stress at solid-liquid interface ($r = W_1$, 1 equation);
- null shear stresses at solid-liquid interface ($r = W_1$, 2 equations);
- null normal and shear stresses at vacuum-solid interface ($r = W_2$, 3 equations).

The above conditions yield a system of seven equations in seven unknowns that, as suggested by Baik at al. [76], can be reduced to six by applying the conditions on normal displacement and normal stress together:

$$\begin{aligned} &\sigma_{sr\theta}\Big|_{r=R_1} = 0 \quad \sigma_{sr\theta}\Big|_{r=R_2} = 0 \quad \sigma_{srz}\Big|_{r=R_1} = 0 \quad \sigma_{srz}\Big|_{r=R_2} = 0 \\ &\frac{\sigma_{lrr}}{u_{lr}}\Big|_{r=R_1} = \frac{\sigma_{srr}}{u_{sr}}\Big|_{r=R_1} \quad \sigma_{srr}\Big|_{r=R_2} = 0. \end{aligned} \tag{2.176}$$

To turn (2.176) into an explicit form, (2.174) and (2.175) should be substituted into (2.28) to find the stress components in cylindrical coordinates as a function of the potentials. Hence for the liquid domain:

$$\hat{\sigma}_{lrr} = -\lambda \frac{\omega^2}{c_\phi^2} \overline{\Phi}_n^\phi e^{in\theta} e^{ik_z z} \tag{2.177a}$$

$$\hat{\sigma}_{lr\theta} = 0 \tag{2.177b}$$

$$\hat{\sigma}_{lrz} = 0, \tag{2.177c}$$

and for the solid domain:

$$\begin{aligned} \hat{\sigma}_{srr} = e^{ik_z z} e^{in\theta} \Big\{ &-\lambda_s \left(q_\gamma^2 + k_z^2\right) \overline{\overline{\Gamma_1 \Gamma_2}}_n^\gamma + 2\mu_s \Big[\Big(\frac{n(n-1)}{r^2} - q_\gamma^2 \Big) \overline{\overline{\Gamma_1 \Gamma_2}}_n^\gamma \\ &+ \frac{q_\gamma}{r} \overline{\overline{\Gamma_1 \Gamma_2}}_{n+1}^\gamma - ik_z q_\psi \overline{\overline{\Psi_1 \Psi_2}}_n^\psi + ik_z \frac{n+1}{r} \overline{\overline{\Psi_1 \Psi_2}}_{n+1}^\psi \\ &+ \frac{n(1-n)}{r^2} \overline{\overline{\Psi_3 \Psi_4}}_n^\psi + \frac{q_\psi n}{r} \overline{\overline{\Psi_3 \Psi_4}}_{n+1}^\psi \Big] \Big\} \end{aligned} \tag{2.178a}$$

$$\hat{\sigma}_{sr\theta} = e^{ik_z z} e^{in\theta} \mu_s \Big\{ \frac{2in(n-1)}{r^2} \overline{\overline{\Gamma_1 \Gamma_2}}^{\gamma}_{n} - \frac{2inq_\gamma}{r} \overline{\overline{\Gamma_1 \Gamma_2}}^{\gamma}_{n+1} - k_z q_\psi \overline{\overline{\Psi_1 \Psi_2}}^{\psi}_{n} + \frac{2k_z(n+1)}{r} \overline{\overline{\Psi_1 \Psi_2}}^{\psi}_{n+1} + (iq_\psi^2 + \frac{2in(1-n)}{r^2}) \overline{\overline{\Psi_3 \Psi_4}}^{\psi}_{n} - \frac{2iq_\psi}{r} \overline{\overline{\Psi_3 \Psi_4}}^{\psi}_{n+1} \Big\} \tag{2.178b}$$

$$\hat{\sigma}_{srz} = e^{ik_z z} e^{in\theta} \mu_s \Big\{ + \frac{2ik_z n}{r} \overline{\overline{\Gamma_1 \Gamma_2}}^{\gamma}_{n} - 2ik_z q_\gamma \overline{\overline{\Gamma_1 \Gamma_2}}^{\gamma}_{n+1} + \frac{q_\psi n}{r} \overline{\overline{\Psi_1 \Psi_2}}^{\psi}_{n} + (k_z^2 - q_\psi^2) \overline{\overline{\Psi_1 \Psi_2}}^{\psi}_{n+1} - \frac{ik_z n}{r} \overline{\overline{\Psi_3 \Psi_4}}^{\psi}_{n} \Big\}. \tag{2.178c}$$

Applying the boundary conditions (2.176) and sorting the equations yields:

$$\begin{bmatrix} d_{11} & d_{12} & d_{13} & d_{14} & d_{15} & d_{16} \\ d_{21} & d_{22} & d_{23} & d_{24} & d_{25} & d_{26} \\ d_{31} & d_{32} & d_{33} & d_{34} & d_{35} & d_{36} \\ d_{41} & d_{42} & d_{43} & d_{44} & d_{45} & d_{46} \\ d_{51} & d_{52} & d_{53} & d_{54} & d_{55} & d_{56} \\ d_{61} & d_{62} & d_{63} & d_{64} & d_{65} & d_{66} \end{bmatrix} \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \\ \Psi_1 \\ \Psi_2 \\ \Psi_3 \\ \Psi_4 \end{bmatrix} = \boldsymbol{D} \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \\ \Psi_1 \\ \Psi_2 \\ \Psi_3 \\ \Psi_4 \end{bmatrix} = 0. \tag{2.179}$$

If the determinant $|\boldsymbol{D}|$ is not zero, the only possible solution is the trivial null solution. However, per each n, it is possible to find a set of values of k_z for which the determinant is null. In that case, the equation admits not null solutions, and the value of all the parameters remains uniquely specified when the initial or additional boundary conditions are given. The value of the coefficients d_{ij} is reported below:

From $\hat{\sigma}_{sr\theta}\Big|_{r=W_1} = 0$:

$$d_{11} = J_n(q_\gamma W_1)\Big[\frac{2in(n-1)}{W_1^2}\Big] + J_{n+1}(q_\gamma W_1)\Big[\frac{-2inq_\gamma}{R_1}\Big] \tag{2.180a}$$

$$d_{12} = Y_n(q_\gamma W_1)\Big[\frac{2in(n-1)}{W_1^2}\Big] + Y_{n+1}(q_\gamma W_1)\Big[\frac{-2inq_\gamma}{W_1}\Big] \tag{2.180b}$$

$$d_{13} = J_n(q_\psi W_1)\Big[-k_z q_\psi\Big] + J_{n+1}(q_\psi W_1)\Big[\frac{2k_z(n+1)}{W_1}\Big] \tag{2.180c}$$

$$d_{14} = Y_n(q_\psi W_1)\Big[-k_z q_\psi\Big] + Y_{n+1}(q_\psi W_1)\Big[\frac{2k_z(n+1)}{W_1}\Big] \tag{2.180d}$$

$$d_{15} = J_n(q_\psi W_1)\Big[\frac{2in(1-n)}{W_1^2} + iq_\psi^2\Big] + J_{n+1}(q_\psi W_1)\Big[-\frac{2iq_\psi}{W_1}\Big] \tag{2.180e}$$

$$d_{16} = Y_n(q_\psi W_1)\Big[\frac{2in(1-n)}{W_1^2} + iq_\psi^2\Big] + Y_{n+1}(q_\psi W_1)\Big[-\frac{2iq_\psi}{W_1}\Big] \tag{2.180f}$$

From $\hat{\sigma}_{srz}\Big|_{r=W_1} = 0$:

$$d_{31} = J_n(q_\gamma W_1)\Big[\frac{2ik_z n}{W_1}\Big] + J_{n+1}(q_\gamma W_1)\Big[-2ik_z q_\gamma\Big] \tag{2.181a}$$

$$d_{32} = Y_n(q_\gamma W_1)\Big[\frac{2ik_z n}{W_1}\Big] + Y_{n+1}(q_\gamma W_1)\Big[-2ik_z q_\gamma\Big] \tag{2.181b}$$

$$d_{33} = J_n(q_\psi W_1)\Big[\frac{q_\psi n}{W_1}\Big] + J_{n+1}(q_\psi W_1)\Big[k_z^2 - q_\psi^2\Big] \tag{2.181c}$$

$$d_{34} = Y_n(q_\psi W_1)\Big[\frac{q_\psi n}{W_1}\Big] + Y_{n+1}(q_\psi W_1)\Big[k_z^2 - q_\psi^2\Big] \tag{2.181d}$$

$$d_{35} = J_n(q_\psi W_1)\Big[-\frac{ik_z n}{W_1}\Big] \tag{2.181e}$$

$$d_{36} = Y_n(q_\psi W_1)\Big[-\frac{ik_z n}{W_1}\Big] \tag{2.181f}$$

Coefficients from $\hat{\sigma}_{sr\theta}\Big|_{r=W_2} = 0$, namely $d_{21} - d_{26}$, can be obtained from $d_{11} - d_{16}$ replacing W_1 with W_2. Coefficients from $\hat{\sigma}_{srz}\Big|_{r=W_2} = 0$, namely $d_{41} - d_{46}$, can be obtained from $d_{31} - d_{36}$ replacing W_1 with W_2.

From $\frac{\hat{\sigma}_{lrr}}{u_{lr}}\Big|_{r=W_1} = \frac{\hat{\sigma}_{srr}}{u_{sr}}\Big|_{r=W_1}$, assuming

$$H = \frac{\rho_l \omega^2}{2\rho_s c_\psi^2} \frac{J_n(q_\phi W_1)}{q_\phi J_{n+1}(q_\phi W_1) - \frac{n}{R_1} J_n(q_\phi W_1)} \tag{2.182}$$

yields:

$$d_{51} = J_n(q_\gamma W_1)\Big[\frac{n(1-n+W_1 H)}{W_1^2} - k_z^2 + \frac{k_\psi^2}{2}\Big] + J_{n+1}(q_\gamma W_1)\Big[-q_\gamma \frac{1+W_1 H}{W_1}\Big] \tag{2.183a}$$

$$d_{52} = Y_n(q_\gamma W_1)\Big[\frac{n(1-n+W_1 H)}{W_1^2} - k_z^2 + \frac{k_\psi^2}{2}\Big] + Y_{n+1}(q_\gamma W_1)\Big[-q_\gamma \frac{1+W_1 H}{W_1}\Big] \tag{2.183b}$$

$$d_{53} = J_n(q_\psi W_1)\left[ik_z q_\psi\right] + J_{n+1}(q_\psi W_1)\left[-ik_z \frac{n+1+W_1 H}{W_1}\right] \tag{2.183c}$$

$$d_{54} = Y_n(q_\psi W_1)\left[ik_z q_\psi\right] + Y_{n+1}(q_\psi W_1)\left[-ik_z \frac{n+1+W_1 H}{W_1}\right] \tag{2.183d}$$

$$d_{55} = J_n(q_\psi W_1)\left[\frac{n(n-1-W_1 H)}{W_1^2}\right] + J_{n+1}(q_\psi R_1)\left[-\frac{n q_\psi}{W_1}\right] \tag{2.183e}$$

$$d_{56} = Y_n(q_\psi W_1)\left[\frac{n(n-1-W_1 H)}{W_1^2}\right] + Y_{n+1}(q_\psi W_1)\left[-\frac{n q_\psi}{W_1}\right] \tag{2.183f}$$

Coefficients from $\hat{\sigma}_{srr}\Big|_{r=W_2} = 0$, namely $d_{61} - d_{66}$, can be obtained from $d_{51} - d_{56}$ imposing $H = 0$ and replacing W_1 with W_2. Interestingly, H is a ratio of acoustic pressure to displacement thus, as seen in section 2.6.3, it is similar to the specific acoustic impedance [89].

Once the geometries and the properties of the materials have been assigned, equation (2.179) should be solved in order to obtain k_z in the desired range of frequency ω and for the desired index n. More details are provided in chapter 4.

2.10 Summary

This chapter reviews the literature and the essential concepts of acoustics that provide the theoretical foundations for the model developed in chapter 4. The analysis of the existing literature offers a thorough assessment of the possible strategies to implement the acoustic model and identifies the gaps to be filled by the novel solution developed. In particular, the required liquid-solid boundary condition and its consequence on the modal orthogonality are discussed. The semi-analytic and the quasi-analytic approach are compared as viable options. Then, from the fundamental equations of the linear elasticity theory, the wave equations for sound propagation in both liquid and solid mediums are derived. Some crucial quantities, such as wavenumber, acoustic intensity, and acoustic impedance, are introduced. In addition, the definitions of phase and group velocity and an explanation of the role they play in the description of dispersive effects are reported. The formulation of the boundary conditions problem is introduced for the setup to be modelled. Finally, the equations underlying the propagation

in cylindrical waveguides are formulated for the rigid and elastic boundary cases. An overview of the modal analysis, the related dispersive effects, and the source modelling is provided for the rigid case and as a reference for the concepts further developed in chapter 4.

Chapter 3

Concepts of machine learning

This chapter introduces a few notions of machine learning that will be used to develop automatic in-pipe event classification. The first part provides some basic concepts about collecting and organising data for typical artificial intelligence tasks. These ideas are recalled in chapter 5, where they are adopted to develop the dataset used in this work. The other sections of the chapter describe the tasks typically involved in machine learning: from data representation to some of the techniques used for the automatic extraction of human-meaningful information. A brief overview of the performance assessment is also provided. The concepts introduced are applied and further expanded in chapter 6. Given the number of theories and solutions in the literature, the topics reported here are only those strictly necessary.

3.1 Data for machine learning

The very first common problem among very different artificial intelligence applications, from web security [144],[145] to social network analysis [146],[147], from facial or object recognition [148],[149] to weather and climate modelling [150],[151], is the collection of a sufficiently large amount of examples to be used as a reference to train and test the developing algorithms.

The *training dataset* is usually employed to determine a set of numeric parameters that characterise a specific model used to make predictions about a given

phenomenon. Since the calculated predictions are strictly related to the data used in the training process, the quality of the data employed naturally assumes a key role in the whole development process. Besides, to compare and assess the effectiveness of different proposed approaches, a further distinct set of data, the *test or evaluation dataset*, is required. In certain cases, the evaluation dataset can be simply obtained by separating a smaller non-overlapping portion from the training dataset. In other circumstances, the requirements for the evaluation dataset are more demanding and preparing useful data might require additional effort. When assessing and comparing different solutions, it is necessary to quantify results towards specific goals, avoiding confusion that might mislead the development effort.

The preparation of good data usually concerns two main aspects: the quality of the data itself and the quality of the metadata associated. The quality of the data refers to the capability to represent the space of all the possible examples for a given application. When the application space is extensively represented, trained models are more likely to generalise to unknown examples. Good quality data is generally provided by the three following conditions [38, p. 149]:

- *coverage*: all the relevant classes to be investigated in the application should be included;
- *variability*: each class should be represented by a set of observations that includes a sufficient number of its possible variations;
- *size*: a large number of observations for training and testing purposes should be included.

In reality, however, the conditions above cannot always be fulfilled. For example, those applications usually referred to as *open-set* include unknown classes that, given their unknown nature, cannot be completely represented by the observations in the training dataset, leading to further classification difficulties [152]. Besides, even when all the observations can be associated with specific classes, some classes might be better covered than others providing an imbalanced representation that can affect the outcomes [153].

The quality of the metadata refers to the quality of the *annotations* that are

associated with the raw data to provide meaningful information for both humans and machines. Depending on the application, annotation can be composed of a rich set of information that can be used to improve the effectiveness of the machine learning technique implemented [154]. The most important and basic form of annotation, however, is a simple label proving the association between an observation (or a portion of it) and a specific class. In general, labels should satisfy the following conditions [38, p. 152]:

- *representation*: in the context of a given application, each label should provide a clear description of the item it is associated with;
- *non-ambiguity*: each label should have clear, unique correspondence to only a class.

The above conditions are critical when labels of the same dataset are defined or assigned by different people. In this case, subjective assessment based on personal experiences might lead to inconsistent or meaningless definitions [155]. Ideally, good annotations provide a rich set of additional information and are both machine and human-readable.

3.1.1 Datasets for sound event detection and classification

Further to the purpose of this work, the dataset shall be regarded as a dataset for *sound event detection and classification*. Although the detection is not explicitly investigated, the dataset synthesised as described in chapters 5 and 6 can also be used for detection tasks. Therefore, when useful, some basic detection concepts are mentioned.

The aim of automatic audio classification and detection is to identify specific events belonging to specific classes in a set of audio recordings gathered in a dataset. Labels associated with events in the audio file are named *weak* (or *audio tag*) if they are associated with the entire recording without any specific timing information. Alternatively, *strong* annotations (figure 3.1) provide information about the beginning (*onset*) and the end (*offset*) of the event. When multiple annotations are allowed to overlap in time, they are called *polyphonic* [156].

Certainly, weakly annotated datasets are easier to collect since the class can be simply specified as belonging to the recordings or not. However, the lack of timing information limits their direct usability only to specific cases. Weakly annotated recording, for example, can be employed for automatic feature extraction or, as it will be shown in chapter 5, to build artificial strongly annotated datasets. It is remarked that strong annotations are often necessary. For instance, the final performance assessment usually requires a precise indication of the time. Furthermore, when sound events exhibit noise-like features, it might be difficult to extract structured discriminating information from a random point of the event. In these cases, understanding the behaviour of the source during the initial and final transient might be the key to correctly distinguishing two different classes. This approach can be adopted only if timing information is available during the training process.

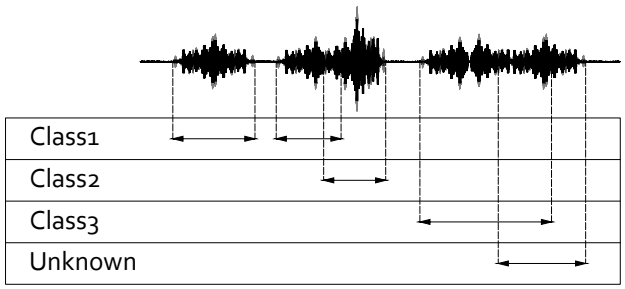


Figure 3.1: *Representation of strong open-set polyphonic annotations for a time domain signal. Class labels are indicated along with the timing information.*

3.2 Signal featurization

The raw representation of a signal in the time domain as a sequence of samples is hardly useful for extracting meaningful information. Basic operations, such as detecting the variation of signal levels, can be performed directly on the raw signal representation, but the amount of information obtainable is very limited. Signal information is generally extracted by comparing sets of signal features with sets of references. This comparison operation is performed by calculating

a certain distance between the former and the latter. When the features used for this calculation are not appropriate, the values of the distances do not allow reliable discrimination between different classes or events. Hence, before applying any classification or event detection algorithm, it is necessary to obtain a useful representation by some kind of featurization process that provides a representation in a well-suited space. Many solutions can be found in the literature, some of which are described in the following sections. Certainly, the effectiveness of each solution depends on the application, and it is one of the key factors for the good performance of the other processing stages. Notwithstanding its importance, the choice of a particular representation rather than a different one is rarely justified [38, p. 96] and some features, such as MFCC (see section 3.2.2), seem to be widely preferred, even beyond their native context [157],[158],[159]. When the extraction of the features is performed with a set of fixed operators, the featurization is usually called *feature engineering*. On the other hand, when the operators include some kind of reference to the processed data, the featurization is called *feature learning*. Feature engineering is usually justified by certain reasons, such as its simplicity or its relation to some bio-inspired equivalent. On the other hand, feature learning can take advantage of the known data to find discriminating signal structures. These two approaches are not necessarily disjoint since engineered features can be used as an input for obtaining learned features [160]. When learned features are obtained without including any reference to the signals metadata, the featurization is said to be *unsupervised*. On the contrary, in the *supervised* case, metadata annotations are used to improve and promote more discriminating features.

3.2.1 Desired feature properties

This section reviews a few desired properties for features used in classification tasks. A quick analysis of techniques commonly used to extract features is further provided.

3.2.1.1 Time-frequency localisation

Acoustic events exhibit peculiar characteristics at both small and large time scales. Smaller time scales, for example, can reveal specific sound textures while

larger time scales can offer an understanding of bigger sound structures and describe temporal dependencies of macro-areas of the signal linked together (e.g. onset, regime, offset) [161],[162]. Unfortunately, despite rendering the best time resolution, a simple analysis in the time domain generally offers poor insight, and better representations are required.

In the following sections, unless otherwise specified, the following definitions of *inner product* for one-dimensional continuous and discrete signals will be used:

$$\langle x(t), y(t) \rangle = \int_{-\infty}^{+\infty} x(t) y^*(t) dt \qquad \langle x[n], y[n] \rangle = \sum_{n=-\infty}^{+\infty} x[n] y^*[n], \tag{3.1}$$

where the * indicates the complex conjugate. The definition of the (squared) *Euclidean norm* follows accordingly as $\|x\|_2^2 = \langle x, x \rangle$. Moreover, the notations $\|x\|_2$ and $\|x\|_1$ indicate respectively $\sqrt{\langle x, x \rangle}$ and $\int_{-\infty}^{+\infty} |x(t)| dt$.

In its general definition, a *linear time-frequency transform* can be seen as a decomposition of a given function in a space defined by a *dictionary* $\mathscr{D}$ of waveforms with unitary norm $\zeta_\gamma(t)$ called *time-frequency atoms* [80, p. 89]. The transform is then given by:

$$\hat{x}_{\mathscr{D}} = \left\{ \langle x(t), \zeta_\gamma(t) \rangle \right\}_{\zeta_\gamma \in \mathscr{D}} = \int_{-\infty}^{+\infty} x(t) \zeta_\gamma^*(t)\, dt \quad \forall\ \zeta_\gamma \in \mathscr{D}. \tag{3.2}$$

To provide certain transformation properties, the functions of the dictionary are usually chosen to be localised in time and frequency. Localisation properties can be defined using mean value and variance. Since $\|\zeta_\gamma(t)\|_2^2 = 1$, the moments in the time domain can be calculated as:

$$\mu_t(\zeta_\gamma) = \int_{-\infty}^{+\infty} t |\zeta_\gamma(t)|^2 dt \qquad \sigma_t^2(\zeta_\gamma) = \int_{-\infty}^{+\infty} (t - \mu_t)^2 |\zeta_\gamma(t)|^2 dt. \tag{3.3}$$

Recalling the *Plancherel's formula* $\|\hat{\zeta}_\gamma(\omega)\|_2^2 = 2\pi \|\zeta_\gamma(t)\|_2^2$ [139, pp. 9.6.13–9.6.14], the equivalent moments in the frequency domain can be written as:

$$\mu_{\frac{\omega}{2\pi}}(\hat{\zeta}_\gamma) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \omega |\hat{\zeta}_\gamma(\omega)|^2 d\omega \quad \sigma^2_{\frac{\omega}{2\pi}}(\hat{\zeta}_\gamma) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} (\omega - \mu_\omega)^2 |\hat{\zeta}_\gamma(\omega)|^2 d\omega. \tag{3.4}$$

Heisenberg's uncertainty theorem [102, p. 288] links together the standard deviations in the time and frequency domain. For the functions of the dictionary $\mathscr{D}$, *Heisenberg's uncertainty relation* holds:

$$\sigma_t \sigma_\omega \geq \frac{1}{2}. \tag{3.5}$$

In the time-frequency plane, the point (μ_t, μ_ω) can be interpreted as the centre of a box of size $\sigma_t \times \sigma_\omega$ — the *Heisenberg's box* — which represents the resolution of the function ζ_γ (figure 3.2).

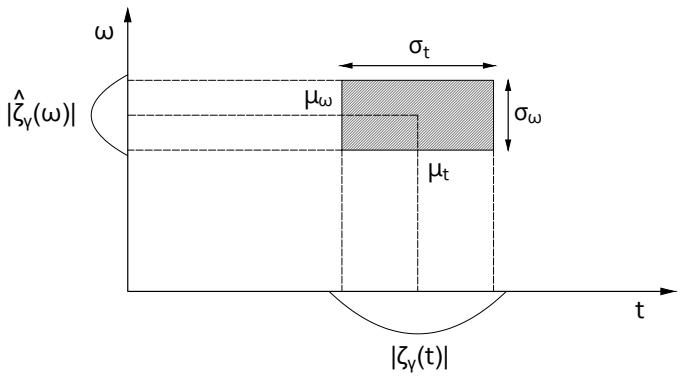


Figure 3.2: *The Heisenberg's box. Position and size of the box are determined by the first and second order moments of* $|\zeta_\gamma(t)|$ *and its Fourier transform* $|\hat{\zeta}_\gamma(\omega)|$*. The resolution is inversely proportional to the box size.*

The relationship 3.5 shows that time resolution cannot be reduced indefinitely without affecting the frequency resolution and vice-versa. Note that, when calculating the time-frequency transform, the coefficient associated with ζ_γ describes the function x around the point (μ_t, μ_ω) in a neighbour proportional to the Heisenberg's box. This can be shown by recalling the Parseval's formulas [139, pp. 9.6.13–9.6.14]:

$$\langle x(t), \zeta_\gamma(t) \rangle = \int_{-\infty}^{+\infty} x(t)\zeta_\gamma^*(t)\,dt = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \hat{x}(\omega)\hat{\zeta}_\gamma^*(\omega)\,d\omega. \tag{3.6}$$

From the previous equation, the value of the transform coefficient associated with ζ_γ is mainly determined from those values of x and $\hat{x}$ where ζ_γ and $\hat{\zeta}_\gamma$ are not negligible, that is, in the Heisenberg's box. Hence, when transforming a signal $x(t)$, the time and frequency resolution of the transformed representation cannot be determined independently. It is also remarked that, to offer a complete representation per each possible signal, each part of the time-frequency plane should be completely covered by the Heisenberg's boxes of the functions in $\mathscr{D}$ [80, p. 19].

Time-frequency localisation of the Fourier transform

Ideally, a perfect representation would have infinite resolution in both the time and frequency domains. From what is shown above, however, this condition is not possible, and the desired representation should balance between time and frequency. The limit cases are given by simple time-domain and Fourier-domain representations, where a set of non-redundant orthonormal functions of the dictionary $\mathscr{D}$ is defined respectively as $\zeta_\eta(t) = \delta(t-\eta)$ and $\zeta_\xi(t) = e^{i\xi t}$ with $\eta, \xi \in \mathbb{R}$. Hence, in the first case, the Heisenberg box will be infinitesimal along the time axis and infinite along the frequency axis, meaning infinite time resolution and null frequency resolution. Conversely, the Fourier transform shows infinite frequency resolution and null time resolution. In both cases, the time-frequency plane is tiled with lines either parallel to the frequency axis (time representation) or parallel to the time axis (frequency representation). Note that the Fourier transform holds the time information in its phase, but since it cannot be easily handled, it is usually discarded. Therefore, considering only the modulus, signals with the same frequency components but completely different time envelopes will appear the same. This is a practical consequence of the null time resolution.

When the signal $x(t)$ is sampled with frequency ω_s, the Fourier transform $\hat{x}(\omega) = \int x(t)e^{-i\omega t}dt$, turns into the *Fourier series*:

$$\hat{x}(\omega) = \int_{-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} x(t)\delta(t-n)e^{-i\omega t}dt = \sum_{n=-\infty}^{+\infty} x[n]e^{-i\omega n}, \tag{3.7}$$

where the spectrum $\hat{x}(\omega)$ is periodic with period equal to the sample frequency ω_s. If $x[n]$ is periodic with a period of finite length N (the periodicity is built by replicating a sequence of finite length N), the dictionary $\mathscr{D}$ can be simply defined with a set of N orthonormal vectors $\zeta_k \in \mathbb{C}^N$ for $\forall\ k \in [0 : N-1]$, where the components of each vector $\zeta_k[n] = e^{\frac{i2\pi kn}{N}}$ are obtained $\forall\ n \in [0 : N-1]$[1]. The *discrete Fourier transform* (DFT) assumes form:

$$\hat{x}[k] = \langle x, \zeta_k \rangle = \sum_{n=0}^{N-1} x[n]e^{\frac{-i2\pi kn}{N}} \qquad with \qquad 0 \leq k < N. \tag{3.8}$$

where $\hat{x}[k]$ has period N. Again, from Parseval's formula for finite discrete signals:

$$\langle x, \zeta_k \rangle = \sum_{n=0}^{N-1} x[n]\zeta_k^*[n] = \frac{1}{N}\sum_{n=0}^{N-1} \hat{x}[n]\hat{\zeta}_k^*[n]. \tag{3.9}$$

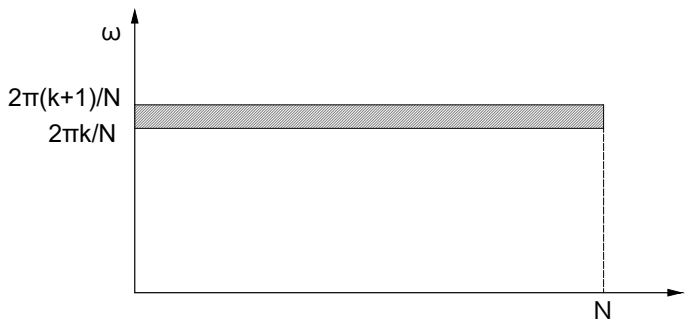


Figure 3.3: *The Heisenberg box of the DFT. The transform provides the coarsest time resolution and the finest frequency resolution.*

For any k, the Heisenberg box of ζ_k is a rectangle of area 2π with length along the time axis equal to N and height equal to the frequency resolution $2\pi/N$ in the interval $\frac{2\pi}{N}[k,(k+1)]$ (figure 3.3).

[1] if $k = h$ $\langle \zeta_k, \zeta_h \rangle = N$
if $k \neq h$ $\langle \zeta_k, \zeta_h \rangle = 1 + r + ... + r^{N-1} = \frac{1-r^N}{1-r} = 0$ with $r = e^{\frac{i2\pi(k-h)}{N}}$

In both the continuous and discrete cases, the Fourier transform exhibits the finest frequency resolution and, at the same time, the coarsest time resolution. The poor time resolution is an undesired feature for signal discrimination, and this issue is the first main limitation of approaches based on the Fourier transform.

Time-frequency localisation of the short-time Fourier transform

A simple solution to balance time and frequency resolution is given by the *short-time Fourier transform (STFT)* [80, pp. 92–101], which can be seen as a localised version of the Fourier transform. In the continuous case, the functions of the dictionary $\mathscr{D}$ can be defined as:

$$\zeta_{\eta,\xi}(t) = \phi(t-\eta)e^{i\xi t} \qquad \forall(\eta,\xi) \in \mathbb{R}^2, \tag{3.10}$$

and the STFT of a signal $x(t) \in \mathbf{L}^2(\mathbb{R})$ is:

$$\hat{x}(\eta,\xi) = \langle x, \zeta_{\eta,\xi} \rangle = \int_{-\infty}^{+\infty} x(t)\phi(t-\eta)e^{-i\xi t}dt \qquad \forall(\eta,\xi) \in \mathbb{R}^2. \tag{3.11}$$

The function $\phi(t)$ is called the *window function*. It is real, even ($\phi(t) = \phi(-t)$), and, to keep the norm of $\zeta_{\eta,\xi}$ unitary, with its norm unitary. $\phi(t)$ localises the function $x(t)$ in time and is usually chosen to provide specific properties such as smooth transitions at the boundaries [163, p.87]. The calculation of the moments for $\zeta_{\eta,\xi}$ yields:

$$\mu_t = \int t|\zeta_{\eta,\xi}(t)|^2 dt = \int t|\phi(t-\eta)|^2 dt = \eta \tag{3.12a}$$

$$\sigma_t^2 = \int (t-\eta)^2|\zeta_{\eta,\xi}(t)|^2 dt = \int (t-\eta)^2|\phi(t-\eta)|^2 dt = \int t^2|\phi(t)|^2 dt \tag{3.12b}$$

$$\mu_\omega = \int \omega|\hat{\zeta}_{\eta,\xi}(\omega)|^2 d\omega = \int \omega|\hat{\phi}(\omega-\xi)|^2 d\omega = \xi \tag{3.12c}$$

$$\sigma_\omega^2 = \int (\omega-\xi)^2|\hat{\zeta}_{\eta,\xi}(\omega)|^2 d\omega = \int \omega^2|\hat{\phi}(\omega)|^2 d\omega, \tag{3.12d}$$

where μ_ω is found by considering that the Fourier transform $\hat{\phi}(\omega)$ is real and

symmetric as $\phi(t)$ and $|\hat{\zeta}_{\eta,\xi}(\omega)| = |\hat{\phi}(\omega-\xi)e^{-i\eta(\omega-\xi)}| = |\hat{\phi}(\omega-\xi)|$ is simply a translation by ξ of $\hat{\phi}(\omega)$ in frequency. From equations (3.12), it appears that Heisenberg's boxes are simply translated in time and frequency by η and ξ and have a constant size determined by the window function $\phi(t)$. Hence, the STFT tiles the time-frequency plane with boxes of constant size and provides the same time-frequency resolution everywhere. Note that, since in the definition (3.10) η and ξ vary continuously in $\mathbb{R}$, each point of the time-frequency plane is covered by an infinite number of Heisenberg's boxes and (3.11) provides a very redundant representation. The geometry of Heisenberg's box can be easily changed by scaling the window function. Indeed, $\phi_s(t) = s^{-1/2}\phi(t/s)$ increases the time length by s and reduces the frequency spread by the same factor, leaving the area of the box unaltered. A non-redundant representation can be obtained by discretizing the shift parameters and defining a time hop η_0, and a frequency hop ξ_0 according to the area occupied by Heisenberg's box. Hence, the functions of the dictionary assume the form:

$$\zeta_{h,m}(t) = \phi_s(t - hs\eta_0)e^{i\frac{m\xi_0}{s}t} \qquad \forall(h,m) \in \mathbb{Z}^2, \tag{3.13}$$

where the scale factor s has also been included. To provide a complete representation, however, further conditions need to be imposed on the window $\phi(t)$ [80, p. 183]. Despite its simplicity and the possibility of reshaping Heisenberg's box, the characteristic of having constant time-frequency resolution might be undesired when certain sound patterns require variable resolution, either time or frequency.

The discrete version of the STFT for a signal $x[n]$ of length N is defined as:

$$\langle x, \zeta_{h,k}\rangle = \sum_{n=0}^{K-1} x[n]\phi[n-hH]e^{\frac{-i2\pi kn}{K}} \qquad 0 \le k < K \quad 0 \le h < H_{max}, \tag{3.14}$$

where K is the length of the window and $H_{max} = N/H$. For simplicity, N/H is assumed integer. Note that, per each h, the STFT matches the DFT of $x[n]\phi[n-hH]$ in the length of the window. In the previous definition, the vectors of the dictionary are chosen as:

$$\zeta_{h,k} = \phi[n - hH]e^{\frac{i2\pi kn}{N}} \qquad \forall\, n = [0 : K-1]. \tag{3.15}$$

The constant $H \leq N$ defines the *hop size*, that is, the shift of the window $\phi[n]$ in time. To evaluate Heisenberg's box, it is observed that the sum (3.14) is non-null only over an interval of length K centred in hH, that is, the support of the shifted window. As for the continuous case, the length along the time axis of the Heisenberg box depends only on the window $\phi[t]$ and does not change with its position in the time-frequency plane. To assess the height, the DFT of $\zeta_{h,k}$ can be written as:

$$\hat{\zeta}_{h,k}[l] = \sum_{n=0}^{K-1} \phi[n - hH]e^{\frac{i2\pi kn}{K}} e^{\frac{-i2\pi ln}{K}} = \hat{\phi}[l-k]e^{\frac{-i2\pi hH(l-k)}{K}}, \tag{3.16}$$

with $\hat{\phi}[k]$ DFT of $\phi[n]$. Again, $\hat{\zeta}_{h,m}$ depends only on the Fourier transform of the window function $\hat{\phi}[k]$ and does not change with the frequency shift k.

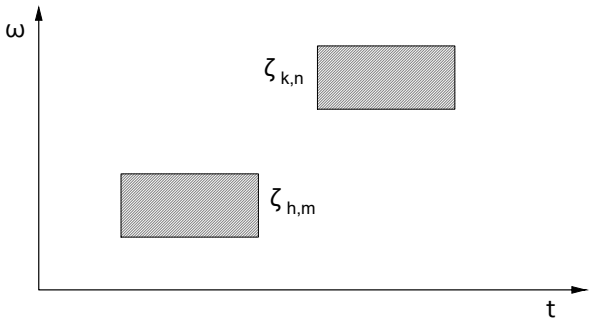


Figure 3.4: *The Heisenberg's box of the STFT. The resolution of STFT is independent on both time and frequency shift.*

Therefore, as for the continuous case, the STFT provides the same time-frequency resolution per each value of time and frequency (figure 3.4). Note that, to offer a complete representation, the following *constant-overlap-add* (*COLA*) condition is usually imposed on the windows [80, p.184]:

$$\sum_{h=0}^{H_{max}-1} |\phi[n-hH]|^2 = A > 0 \quad \forall n \in [0:N-1]. \tag{3.17}$$

In chapter 5, the discrete STFT will be used for the noise filters, and the previous condition will be used to reverse the transform and recover the signal $x[t]$.

A commonly adopted trade-off for audio signals is a window of $20 \div 50ms$ [164], [38, p. 27]. This is usually thought of as the maximum length beyond which the benefit of higher frequency resolution does not compensate for the loss of temporal details.

3.2.1.2 Translation invariance

When analysing audio or images, translation is unavoidable. The simplest case is an identical audio signal whose start time has been delayed. Obviously, if the purpose is to characterise certain information included in the signal, the features extracted should guarantee *invariance* to the translation. A less stringent but still useful property is the *equivariance*, which means that the extracted features will also be translated without changing their values. Formally, the time-frequency transform of the shifted signal $\langle x(t-\mu), \zeta_\gamma(t)\rangle$ will return a shifted representation $\langle x(t), \zeta_\gamma(t+\mu)\rangle$, if $\zeta_\gamma(t+\mu)$ still belongs to the dictionary $\mathscr{D}$ up to a multiplicative constant [80, p. 91]. Hence, equivariant dictionaries can be built by translating a family of generator functions. Invariance and equivariance are often necessary properties in classification tasks and play a key role in the definition of the dictionaries for time-frequency representations [165],[166].

The modulus of the Fourier transform, for example, is translation invariant since:

$$|\hat{x}_\eta(\omega)| = \left| \int_{-\infty}^{+\infty} x(t-\eta) e^{-i\omega t} dt \right| = |e^{-i\eta\omega}\hat{x}(\omega)| = |\hat{x}(\omega)|. \tag{3.18}$$

The STFT is translation invariant if the shift η is negligible with respect to the length of the window. If the shift is a multiple of the hop size, it is possible to obtain at least the equivariance. From this point of view, choosing a window sufficiently small, as in the previous section, always guarantees equivariance in practice.

Another possible invariant representation is given by the *autocorrelation* whose invariance follows directly from the definition:

$$Cx(u) = \int_{-\infty}^{+\infty} x(t)x(t-u)dt. \tag{3.19}$$

In general, at the cost of losing signal information, it is possible to transform a time domain representation into some low-dimensional translation invariant features. For example, *zero-crossing rate*, which for discrete signal is defined as:

$$Z_{cr}x = \frac{\sum_{n=2}^{N} |sign(x[n]) - sign(x[n-1])|}{2(N-1)}, \tag{3.20}$$

are translation invariant but discard almost all the information associated with the signal. Similarly, statistical moments:

$$\mu = \mathbb{E}[x] \tag{3.21}$$

$$\sigma^2 = \mathbb{E}[(x-\mu)^2] \tag{3.22}$$

$$s_n = \frac{\mathbb{E}[(x-\mu)^n]}{\sigma^n} \tag{3.23}$$

such as *mean* μ $(n=1)$, *variance* σ^2 $(n=2)$, *skewness* s_3 (temporal asymmetry) $(n=3)$, and *kurtosis* s_4 (temporal flatness) $(n=4)$ provide only a very coarse interpretation of the signal properties and are affected by accuracy issues increasing with the order of the moments [167]. Similarly, a time-frequency representation that is not invariant but only equivariant can be turned invariant by performing some averaging operations on the extracted features [168]. The drawback is, again, the loss of information potentially relevant.

3.2.1.3 Stability to deformations

In real applications, different instances of the same class might exhibit a certain degree of deformation. An image marginally distorted and a slightly tuned tone are just two simple examples. In general, when the deformation is small, it is desirable to obtain features whose values are close to the original ones or, at least, values

that change proportionally to the deformation. Unfortunately, this *stability* property does not belong to all the representations, and it is certainly something to account for when choosing a certain solution for the features. Obviously, deformations can appear in different flavours, and some transforms that are stable in respect of certain deformations might behave differently with respect to others. Some cases, however, are particularly meaningful and allow some important discrimination between transforms. In [82], it is pointed out that, for a translation-equivariant nonexpansive transforms[2] $\{\zeta\}$, the stability can be characterised using *Lipschitz continuity* relative to small deformations close to translations. It means that, if the signal $x(t)$ is deformed by $\tau(t)$ as $x_\tau(t) = x(t - \tau(t))$ with $|\tau'(t)| < 1$, then the transformation $\{\zeta\}$ is stable if:

$$\|\{\zeta(x)\} - \{\zeta(x_\tau)\}\|_2 \leq C \sup_t |\tau'(t)| \|x\|_2, \tag{3.24}$$

that is, if the transform $\{\zeta\}$ is Lipschitz-continuous in respect to the deformation τ. The constant $C > 0$ can be interpreted as a measure of stability. Note that since Lipschitz continuity guarantees differentiability almost everywhere, the deformations are locally linearized by the transform $\{\zeta\}$. Therefore, a family of small deformations generates a linear space, and invariants to these deformations can be found in a space built as its orthogonal complement [79].

The action of the deformation $\tau(t)$ can be exemplified by considering a little shift in the tones emitted by a mechanical sound generator. This might be caused by unavoidable tweaks in the mechanical setup. These differences, however, shouldn't be relevant for the purpose of sound classification. A simple model to describe this kind of variation for a signal $x(t)$ can be obtained by assuming $\tau(t) = \varepsilon t$ with ε small. If the features are extracted using the Fourier transform, the representation of $x_\varepsilon(t) = x(t - \varepsilon t)$ can be written as $\hat{x}_\varepsilon(\omega) = \frac{1}{1-\varepsilon}\hat{x}(\frac{\omega}{1-\varepsilon})$. Hence, the frequency shift $|\omega - \omega/(1-\varepsilon)|$ is proportional to the frequency of the tones and is more relevant at higher frequencies. If high-frequency tones have a sufficiently high amplitude, $\|\hat{x}(\omega) - \hat{x}_\varepsilon(\omega)\|_2$ does not diminish with ε since the two spectra do not

[2] Nonexpansive means: $\|\{\zeta(f)\} - \{\zeta(h)\}\|_2 \leq \|f - h\|_2$.

overlap where most of the energy is located. Hence, the Fourier transform does not satisfy the 3.24. It is remarked that this behaviour is certainly related to the fine frequency resolution of the Fourier transform. Indeed, if a transform operated some frequency averaging across large frequency bands, the abovementioned issue would be mitigated. From this point of view, the STFT works better than simple FT, but it does not solve the problem. In fact, dealing with Heisenberg's boxes of fixed size introduces an unavoidable trade-off between the averaging operation (generally more relevant at higher frequencies) and the necessity to preserve the lower spectrum (where it is usually necessary to distinguish tones whose frequencies are much closer).

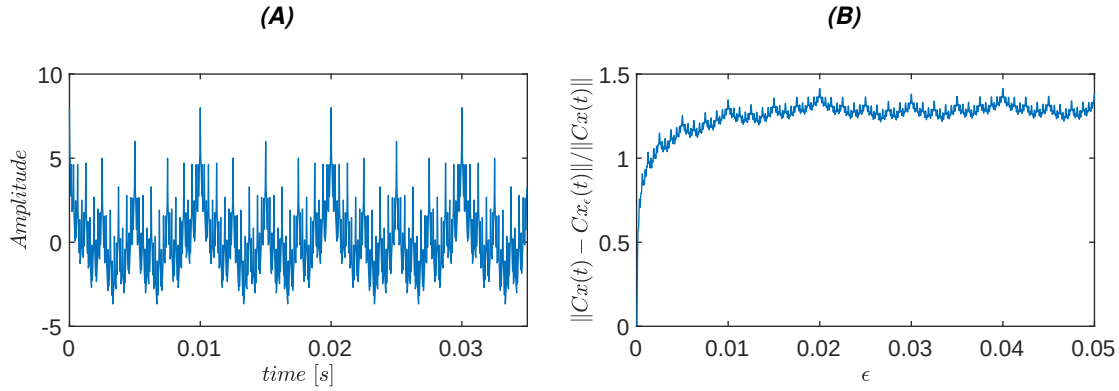


Figure 3.5: *A stationary signal $x(t)$ obtained as a sum of eight cosine tones placed at intervals of one octave starting from $100Hz$ (left). The norm of the distance between the autocorrelation of $x(t)$ and the autocorrelation of the deformed version $x_\varepsilon(t)$ (right). Distance blows for very small values of ε and saturates when the representations do not overlap.*

Another frequently used [169],[170] time-invariant operator, the auto-correlation, suffers the same kind of instability. In fact, since the Fourier transform of the auto-correlation of $x(t)$ is $|\hat{x}(\omega)|^2$, applying the Plancherel formula yields $\|Cx - Cx_\tau\|_2^2 = \frac{1}{2\pi}\||\hat{x}(\omega)|^2 - |\hat{x}_\tau(\omega)|^2\|_2^2$. Figure 3.5A illustrates a signal $x(t)$ obtained by summing a number of cosines tones placed at intervals of one octave starting from $100Hz$. Figure 3.5B reports the norm of the distance $\|Cx(t) - Cx_\varepsilon(t)\|_2^2$ with $\tau(t) = \varepsilon t$ for different values of ε. Cx is calculated as circular auto-correlation. As shown, the distance becomes immediately very large for very small values of ε and saturates

when $Cx(t)$ and $Cx_\varepsilon(t)$ do not overlap. In a stable representation, the norm of the distance would change roughly linearly with the deformation.

3.2.2 MFCC

Many signal representations found in the literature are based on ideas that come from bio-acoustics research. Some non-exhaustive examples are *critical bands* [171], *Bark scale* [172] and *gammatone filters* [173]. In the context of machine learning, the most widely used features, the *Mel Frequency Cepstral Coefficients* (*MFCC*), are based on the *mel scale*. The mel scale is a transformation of the linear frequency scale into a different one mapped on the psychological sensations of pure tones [174]. Unfortunately, since the measurements are based on subjective perceptions, more than a single definition can be found in the literature. A popular formula, mentioned for example in [175], reports:

$$\bar{f} = 2595 \times log_{10}\left(1 + \frac{f}{700}\right), \tag{3.25}$$

while the formula reported in [176, p. 19]:

$$\bar{f} = \frac{f_0}{log(C)} \times log\left(1 + \frac{f}{f_0}\right) = \frac{1000}{log(2)} \times log\left(1 + \frac{f}{1000}\right). \tag{3.26}$$

In the previous equation, the term $log(C)$ provides a Mel-Hz conversion not dependent on the particular logarithmic base, while the *corner frequency* f_0 is experimentally determined. Usually, the lowest part of the spectrum is discarded since sources such as voices exhibit negligible energy in that range and the typical $50Hz$ 'hum' can be avoided.

Given an input signal $x(t)$, to calculate the MFCC, firstly, the energy of the *spectrogram* $|\hat{x}(t,f)|^2$ is obtained as the squared modulus of the STFT $\hat{x}(t,f)$. The STFT is calculated with a window $\phi(t)$ of length T. Then, the energy is averaged over several frequency bands using a set of filters $\{\hat{\psi}_\lambda(f)\}$ in a *filter bank*. In the next step, the amplitude non-linearity of the auditory system is modelled by taking the log_{10} of the averaged energies. Since filters overlap, the

resulting filtered energies are correlated. To decorrelate, in the last step, the MFCC coefficients are calculated using the *cosine transform*.

Further details about the MFCC can be found in [177]. Here it is useful to analyse the frequency averaging operation performed by the filter bank since there lies the reason for the effectiveness of these standard features. Mel filter banks are obtained by dividing the desired frequency range in the Mel scale into $P+1$ equal intervals. Let $[\bar{f}_0, \bar{f}_1, ..., \bar{f}_{(P+1)}]$ be the $P+2$ boundaries of the intervals. The distance between adjacent points is $D_m = (\bar{f}_{(P+1)} - \bar{f}_0)/(P+1)$. In the Mel scale, the frequency support of the λ^{th} filter has length $2D_m$ and center in $\bar{f}_\lambda$ with $\lambda = [1:P]$. Back to the linear frequency scale, the ratio of the frequency band to the central frequency for the λ^{th} filter is:

$$\frac{1}{Q} = \frac{f_{\lambda+1} - f_{\lambda-1}}{f_\lambda} = \frac{\left(C^{\frac{\bar{f}_{\lambda+1}}{f_0}} - 1\right) - \left(C^{\frac{\bar{f}_{\lambda-1}}{f_0}} - 1\right)}{C^{\frac{\bar{f}_\lambda}{f_0}} - 1} \approx C^{\frac{D_m}{f_0}} - C^{-\frac{D_m}{f_0}}, \qquad (3.27)$$

which does not depend on the filter index λ when $C^{\frac{\bar{f}_\lambda}{f_0}} \gg 1$. Hence, accounting for the approximations, the filter bank $\{\psi_\lambda\}$ can be implemented as a set of *Q-constant* filters where each filter $\hat{\psi}_\lambda(f)$ is centred in f_λ.

In practical applications, the filters ψ_λ are taken so that $|\hat{\psi}_\lambda(f)|^2$ is triangular, and the output in the λ^{th} band is calculated as a weighted sum of energy components. The weights are given by the corresponding amplitude of $|\hat{\psi}_\lambda(f)|^2$. For a typical bandwidth of $16kHz$, the number of filters is commonly chosen between 12 and 30 [38, p. 81]. A common filter bank frequency response in the linear frequency scale is reported in figure 3.6A

From above, integrating in ω, the averaged energy at the output of the filter ψ_λ can be written as:

$$Mx(t,\lambda) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} |\hat{x}(t,\omega)|^2 |\hat{\psi}_\lambda(\omega)|^2 d\omega. \qquad (3.28)$$

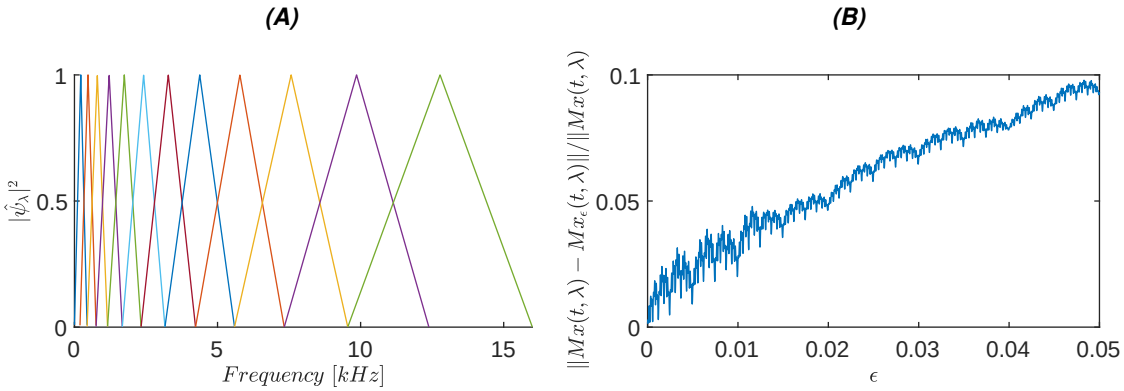


Figure 3.6: *A normalised filter bank of 12 triangular filters $|\hat{\psi}_\lambda(f)|^2$ over a frequency range of $16kHz$ (A). The norm of the difference between the MFCC representation of $x(t)$ and the MFCC representation of $x_\varepsilon(t)$ (B). $x(t)$ is the same signal reported in figure 3.5A. To reduce the effect of the side lobes of the filter, a Chebyshev window has been used for $\phi(t)$. Since $x(t)$ is stationary, the window length does not play a very relevant role in this case. Note that the distance increases linearly as expected, and it is more than an order of magnitude smaller than for the autocorrelation.*

From the definition of the STFT (3.11), recalling that $\hat{x}(t,\omega)$ is the Fourier transform of $x_t = x(u)\phi(u-t)$, (3.28) turns in:

$$Mx(t,\lambda) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} |\hat{x}_t(\omega)|^2 |\hat{\psi}_\lambda(\omega)|^2 d\omega. \tag{3.29}$$

Applying Plancherel's formula yields:

$$\begin{aligned} Mx(t,\lambda) &= \int_{-\infty}^{+\infty} |x_t(v) \star \psi_\lambda(v)|^2 dv \\ &= \int_{-\infty}^{+\infty} \left| \int_{-\infty}^{+\infty} x(u)\phi(u-t)\psi_\lambda(v-u)du \right|^2 dv, \end{aligned} \tag{3.30}$$

where the product in the Fourier domain has been turned into the convolution $\star$ in the time domain. Recalling that the MFCC are not defined for low frequencies, it is possible to assume the band of the λ^{th} filter ω_λ/Q so that $T > 2\pi Q/\omega_\lambda$, where T is the duration of the window $\phi(t)$. Hence, the window $\phi(t)$ is roughly constant on the support of $\psi_\lambda(t)$ and it yields:

$$
\begin{aligned}
Mx(t,\lambda) &\approx \int_{-\infty}^{+\infty} \left| \int_{-\infty}^{+\infty} x(u)\phi(v-t)\psi_\lambda(v-u)du \right|^2 dv \\
&= \int_{-\infty}^{+\infty} \left| \int_{-\infty}^{+\infty} x(u)\psi_\lambda(v-u)du \right|^2 |\phi(t-v)|^2 dv \\
&= |x \star \psi_\lambda|^2 \star |\phi|^2(t),
\end{aligned}
\tag{3.31}
$$

where $\phi(t) = \phi(-t)$ has also been used. From the previous equation, the calculation of the MFCC at frequency ω_λ involves the convolution with two different filters: $\psi_\lambda(t)$ and $|\phi(t)|^2$.

The band-pass filter $\psi_\lambda(t)$ provides stability to deformations by operating frequency averaging over its own band. Recalling the time-warping deformation described in 3.2.1.3, the progressively larger shift proportional to the frequency of the tone $|\omega - \omega/(1-\varepsilon)| = |\omega|\varepsilon$ is balanced by filters that have a progressively larger band $\omega_\lambda/Q \approx |\omega|/Q$. This guarantees that a tone and its deformed copy are described by the same coefficients (figure 3.6B).

The window function $|\phi(t)|^2$ can be seen as a low pass filter that averages $|x(t) \star \psi_\lambda(t)|^2$ over the length T of the window. This time-averaging operation introduces time invariance as from section 3.2.1.2 but, at the same time, discards most of the information associated with $|x(t) \star \psi_\lambda(t)|^2$. This issue limits the performance of the MFCC, and it is one of the main reasons behind the introduction of the *wavelet scatting transform* (see section 3.2.3.4) [79]. In practical applications, the length T for the MFCC is chosen by balancing the loss of information with the time-invariance and the need to characterise longer sound structures. As for the STFT, T is commonly chosen in the range $20 \div 50ms$.

3.2.3 Wavelets representations

In the previous section, it has been shown how MFCC introduces stability to time-warping deformations and time invariance by approximately performing a frequency averaging filtering and a time averaging filtering. Unfortunately, by seeking time-invariance, a certain amount of information associated with the signal is discarded, and this issue generally limits the performance of classification tasks.

A solution to this issue comes from *wavelets*. In particular, the *wavelet scattering transform* offers a strategy not to lose the discarded information without affecting invariance and time-warping stability. Wavelets have been rigorously formalised in literature relatively recently [178], but their first application dates back in time. Wavelets find applications in a wide range of fields, from physics [179] to medicine [180]. In machine learning, they are frequently applied to extract sets of descriptive features, for example, for audio [181] and images [182]. The introduction of the discrete scatting transform (section 3.2.3.4) has also offered some interesting insight into the reasons behind the good performance of deep neural networks in artificial intelligence tasks [183]. Here, only a short review of the main concepts is given, providing only those concepts that are used in chapter 6 for detection and classification tasks.

3.2.3.1 Real and analytic wavelets

A function $\psi(t) \in \mathbf{L}^2(\mathbb{R})$[3] centered around $t = 0$, with zero average and unitary norm:

$$\mu_\psi = \int_{-\infty}^{+\infty} \psi(t)dt = 0 \qquad \|\psi\|_2^2 = \int_{-\infty}^{+\infty} |\psi(t)|^2 dt = 1 \tag{3.32}$$

is called a *wavelet* [80, p. 102]. A dictionary of wavelets is generated by scaling and translating a *mother wavelet* $\psi(t)$ while keeping (3.32) valid for each atom of the dictionary:

$$\mathscr{D}_\psi = \{\psi_{u,s}(t)\} = \left\{ \frac{1}{\sqrt{s}} \psi\left(\frac{t-u}{s}\right) \right\}_{u\in\mathbb{R}, s\in\mathbb{R}^+}. \tag{3.33}$$

The definition of the wavelet dictionary is usually supplemented with the *admissibility condition*:

$$\int_0^{+\infty} \frac{|\hat{\psi}(\omega)|^2}{\omega} d\omega < +\infty. \tag{3.34}$$

For $x(t) \in \mathbf{L}^2(\mathbb{R})$, (3.34) provides a sufficient condition [80, p. 105-110] for the

[3] $\mathbf{L}^2(\mathbb{R})$ indicates the functions square-integrable on $\mathbb{R}$.

continuous wavelet transform $Wx(u,s) = \{\langle x(t), \psi_{u,s}(t)\rangle\}_{u\in\mathbb{R},s\in\mathbb{R}^+}$ to be complete (any function $x(t) \in \mathbf{L}^2(\mathbb{R})$ can be recovered from its transform), and to exhibit energy conservation (the energy of $x(t)$, $\|x\|_2^2 = \int_{-\infty}^{+\infty} |x(t)|^2 dt$, can be recovered from the energy of its transform). Note that, to guarantee (3.34), it is sufficient that the Fourier transform of $\psi(t)$ is null in $\omega = 0$ ($\hat{\psi}(0) = 0$), and it is either continuously differentiable or sufficiently fast decaying:

$$\int_{-\infty}^{+\infty} (1+|t|)|\psi(t)|dt < +\infty. \tag{3.35}$$

Apart from the conditions above, no further restriction is given on the nature of the function $\psi(t)$, which usually depends on the particular application. A first important distinction can be made between *real* and *complex analytic wavelets*. When the function $\psi(t)$ is real, the wavelet transform $Wx(u,s)$ is used to measure the transitions of $x(t)$ in an interval around u proportional to s. On the other hand, choosing $\psi(t)$ complex and analytic[4], allows to separate amplitude and phase components, giving the means to measure evolution in time of frequency structures [80, p. 103].

A possible way to define an analytic mother wavelet is to choose $\psi(t) = e^{i\xi t}\theta(t)$ where $\xi > 0$ and the function $\theta(t)$ is real and even ($\theta(\omega) = \theta(-\omega)$) and is taken so that $\hat{\theta}(\omega) = 0$ for $|\omega| > \xi$. ξ allows setting the frequency shift as for the

[4] To be analytic, the Fourier transform of a function $f_a(t)$ must be null for negative frequencies:

$$\hat{f}_a(\omega) = 0 \qquad \forall \omega < 0. \tag{3.36}$$

Given its Fourier transform, $f_a(t)$ is necessarily complex, but it can be entirely characterised by its real part $f(t)$. In the time domain, this dependence can be written as:

$$f_a(t) = f(t) + iH(f(t)) = A(t)e^{i\phi(t)} \tag{3.37}$$

where $H(f(t)) = \frac{1}{\pi t} \star f(t)$ is the Hilbert transform of $f(t)$ [184, p. 38]. Note that the previous equation can also be used to obtain the analytic version of a real signal $f(t)$. From (3.37), to obtain a real function $f(t)$ from its analytic version $f_a(t)$, it is sufficient to take the real part of $f_a(t)$. Hence, $f(t)$ can also be written as:

$$f(t) = \Re(f_a(t)) = A(t)cos(\phi(t)) \tag{3.38}$$

where the identity $e^{ix} = \cos x + i\sin x$ has been used. Therefore, using its analytic version, a real function $f(t)$ can be decomposed into an *envelope* $A(t) = |f_a(t)|$ and a *carrier* $cos(\phi(t))$ with $\phi(t) = \arg(f_a(t))$ [185]. The meaning of $A(t)$ and $\phi(t)$ depends on the signal $f(t)$. Note that when $f(t)$ is narrow-band, e.g. an AM modulated signal with a single frequency carrier, taking the modulus of the analytic version extracts the envelope, which matches the modulated signal. The envelope has its own frequency spectrum, and in general, calculating the modulus of $|f_a(t)|$ results in a shift to lower frequencies of the resulting spectrum.

STFT, and the previous operation corresponds to the frequency modulation of a low pass filter. Since $\hat{\psi}(\omega) = \hat{\theta}(\omega - \xi)$ and has null spectrum for $\omega < 0$, $\psi(t)$ is certainly analytic. Besides, given that $\theta(t)$ is real and even, its spectrum is real and symmetric, and the wavelet spectrum is centred in ξ. All the generated wavelets are also analytic since their Fourier transform can be written as:

$$\begin{aligned}\hat{\psi}_{u,s}(\omega) &= \mathscr{F}\left\{\frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)\right\} \\ &= \frac{1}{\sqrt{s}}\int_{-\infty}^{+\infty} e^{i\xi\frac{t-u}{s}}\theta\left(\frac{t-u}{s}\right)e^{-i\omega t}dt = \sqrt{s}\,e^{-i\omega u}\hat{\theta}(s\omega - \xi),\end{aligned} \tag{3.39}$$

which is centered in ξ/s and still null for $\omega < 0$. For instance, a *Morlet* (or *Gabor*) wavelet is obtained with a Gaussian window [80, p. 111]:

$$\theta(t) = \frac{e^{\frac{-t^2}{2\sigma_N^2}}}{\sigma_N\sqrt{2\pi}} \qquad \hat{\theta}(\omega) = e^{-\frac{\sigma_N^2\omega^2}{2}}, \tag{3.40}$$

where the previous definition has been normalised to obtain $\max(\hat{\theta}(\omega)) = 1$. Note that $\theta(t)$ and $\hat{\theta}(\omega)$ are both Gaussian and the product of their standard deviation in the time and the Fourier domain is unitary. The Morlet wavelets are only approximately analytic (*pseudo-analytic*) since the spectrum is not exactly null for negative frequencies. However, their spectrum for $\omega < 0$ can be forced to 0 by weakening the constraints about the decay in the time domain.

Unless otherwise specified, the following sections refer to analytic wavelets since they are functional to the signal featurization used in 6.

3.2.3.2 Wavelets time-frequency localisation

In section 3.2.2 it has been shown that Q-constant filter banks exhibit frequency resolution decreasing with the central frequency of the filter, and this feature is crucial to guarantee stability to time-warping deformations. Wavelets reveal a similar characteristic, and a simple calculation of the Heisenberg box is provided in this section to illustrate this point.

As seen in section 3.2.1.1, the time-frequency localisation of the wavelet transform $Wx(u,s) = \{x(t), \psi_{u,s}(t)\}$ depends on time and frequency spread of the atoms of the dictionary. If the mother wavelet is assumed to be time-centred in $t = 0$ as for the Morlet wavelets, the time-centre of the atom is $\mu_t = u$. The time-spread σ_t is:

$$\begin{aligned}\sigma_t^2 &= \int_{-\infty}^{+\infty} (t-u)^2 |\psi_{u,s}(t)|^2 dt \\ &= \int_{-\infty}^{+\infty} (t-u)^2 \Big| \frac{1}{\sqrt{s}} \psi\Big(\frac{t-u}{s}\Big)\Big|^2 dt = s^2 \int_{-\infty}^{+\infty} \underline{t}^2 |\psi(\underline{t})|^2 d\underline{t} = s^2 \bar{\sigma}_t^2,\end{aligned} \tag{3.41}$$

where $\bar{\sigma}_t^2 = \int_{-\infty}^{+\infty} t^2 |\psi(t)|^2 dt$ is the time spread of the mother wavelet. As seen in the previous paragraph, for a mother wavelet frequency-centred in:

$$\xi = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \omega |\hat{\psi}(\omega)|^2 d\omega, \tag{3.42}$$

the generated wavelets are frequency-centred in $u_\omega = \xi/s$. The frequency spread can be calculated as:

$$\begin{aligned}\sigma_\omega^2 &= \frac{1}{2\pi} \int_{-\infty}^{+\infty} \Big(\omega - \frac{\xi}{s}\Big)^2 |\hat{\psi}_{u,s}(\omega)|^2 d\omega \\ &= \frac{1}{2\pi} \int_{-\infty}^{+\infty} \Big(\omega - \frac{\xi}{s}\Big)^2 |\sqrt{s}\, \hat{\psi}(s\omega) e^{-i\omega u}|^2 d\omega \\ &= \frac{1}{s^2} \frac{1}{2\pi} \int_{-\infty}^{+\infty} (\underline{\omega} - \xi)^2 |\hat{\psi}(\underline{\omega})|^2 d\underline{\omega} = \frac{\bar{\sigma}_\omega^2}{s^2},\end{aligned} \tag{3.43}$$

where $\bar{\sigma}_\omega^2$ is the frequency spread of the mother wavelet. The relationships above indicate that the scale parameter s plays a key role in the time-frequency resolution. In particular, a larger scale dilates the wavelet in time, increases the time spread and reduces the frequency spread. Hence, a large s entails high-frequency resolution and low time resolution. Vice versa, a small value of s returns high time resolution and low-frequency resolution. Note that the area of the Heisenberg box $s\bar{\sigma}_t \times \frac{1}{s}\bar{\sigma}_\omega$ always remains the same despite being reshaped. The position of the Heisenberg box depends on both u and s. Figure 3.7 illustrates

two different Heisenberg boxes for two different values of u and s. Note that, with u and s continuous, each point in the time-frequency plane is included in an infinite number of Heisenberg boxes, and the wavelet transform returns a redundant representation.

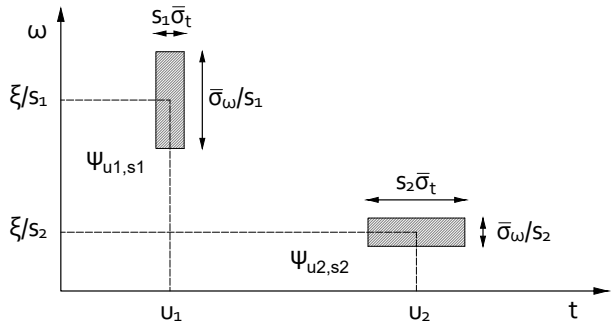


Figure 3.7: *The Heisenberg box of the wavelet transform. Time and frequency resolution depend on the scale value s. Small s means high time resolution and low frequency resolution. Large s means low time resolution and high frequency resolution. The scale s also controls the position of the box along the frequency axis, while the position along the time axis depends only on u.*

A more formal definition is reported in [80, p. 106] where redundancy is characterised by expressing the coefficients of the transform as a function of other coefficients. A non-redundant wavelet representation can be built by sampling time shift and frequency scale to generate a dictionary of orthogonal wavelets[5]. For instance, in [186], an orthogonal wavelet basis is built as:

$$\mathscr{D} = \left\{ \psi_{j,n}(t) = \frac{1}{\sqrt{2^j}} \psi\left(\frac{t - 2^j n}{2^j} \right) \right\}_{(j,n) \in \mathbf{Z}^2} \quad , \tag{3.45}$$

where wavelets are dilated by discrete factors $s = 2^j$ and translated by discrete steps $u = n2^j$. Note that, in this case, the time-frequency plane is tiled by Heisenberg boxes placed at discrete steps. Further completeness and redundancy

[5] A wavelet $\psi_{n,j}(t)$ is orthogonal to another wavelet $\psi_{m,k}(t)$ if:

$$\langle \psi_{n,j}, \psi_{m,k} \rangle = \delta(n,m)\delta(j,k) \tag{3.44}$$

conditions are reported in the literature [80, p. 155].

Calculating the inner product of a translated wavelet can be interpreted as a convolution product. Indeed:

$$\langle x, \psi_{u,s} \rangle = \int_{-\infty}^{+\infty} x(t) \frac{1}{\sqrt{s}} \psi^* \left(\frac{t-u}{s} \right) dt = x \star \overline{\psi}_s(u), \tag{3.46}$$

which shows that the wavelet transform is a convolution between $x(t)$ and the filters $\overline{\psi}_s(t) = \frac{1}{\sqrt{s}} \psi^* \left(\frac{-t}{s} \right)$. Besides, since:

$$\hat{\overline{\psi}}_s(\omega) = \sqrt{s} \hat{\psi}^*(s\omega) \qquad and \qquad \hat{\psi}(0) = \int_{-\infty}^{+\infty} \psi(t) dt = 0, \tag{3.47}$$

then $\hat{\overline{\psi}}_s(0) = 0$, and $\overline{\psi}_s$ can be seen as a bandpass filter.

3.2.3.3 Matching MFCC and wavelet transform

In the previous section, it has been shown that wavelets can be seen as bandpass filters. Besides, if a wavelet in (3.33) is scaled by a factor s, as seen in section 3.2.3.2, both its frequency band and its centre frequency are scaled by a factor $1/s$. This means that a family of wavelets can be seen as a Q-constant filter bank. It is possible to define the desired number M of wavelets per octave by choosing the scale factor as $s = 1/2^{\lambda/M}$ with $\lambda \in \mathbb{Z}$. If the central frequency of the mother wavelet $\hat{\psi}(\omega)$ is ω_c, the generated wavelets will be frequency-centered in $\omega_\lambda = \omega_c 2^{\lambda/M}$. If the mother wavelet is chosen as the one centred at unitary frequency, the centres of the generated wavelets can be written as $\omega_\lambda = 2^{\lambda/M}$. This frequency-normalised mother wavelet is indicated as ψ_0. Since $\mathscr{F}\left(\psi_\lambda(t)\right) = \mathscr{F}\left(\sqrt{\omega_\lambda}\psi_0(\omega_\lambda t)\right) = \frac{1}{\sqrt{\omega_\lambda}} \hat{\psi}_0(\omega/\omega_\lambda)$, to avoid scaling the modulus in the frequency domain, each generated wavelet is multiplied by a scaling factor $\sqrt{\omega_\lambda}$, that is:

$$\psi_{u,\lambda}(t) = \omega_\lambda \psi_0\big(\omega_\lambda (t-u)\big). \tag{3.48}$$

Here, the bandwidth B_0 of ψ_0 is chosen to be $1/M$ to cover the whole frequency

axis. With this choice, $Q = \omega_0/B_0 = M$ and the central frequency of the filters can be written as $\omega_\lambda = 2^{\lambda/Q}$. For instance, for the Morlet wavelets, f_0 and B_0 can be imposed by defining $\xi = 1$ and $\sigma_N = 1/Q$ in (3.39) and (3.40), respectively.

Apart from the filter shape, the family of wavelets described above is almost equivalent to the Q-constant filter bank described for the MFCC. As from equation (3.31), however, calculating MFCC also involves a convolution with the low pass filter $\phi(t)$ obtained from the window function of length T. In section 3.2.2, it has been pointed out that (3.31) is valid for $T > \frac{2\pi Q}{\omega_\lambda}$, that is, for those wavelets ψ_λ whose time spread $\sigma_{t_\lambda} = \frac{2\pi Q}{\omega_\lambda}$ is smaller than the window length T. Hence, the construction of a Q-constant wavelet filter bank is kept only for $\omega_\lambda \geq 2\pi Q/T$ or, equivalently, $\lambda \geq Q\log_2(2\pi Q/T)$. For lower frequencies, in order not to exceed the length T, the spectrum between 0 and $2\pi Q/T$ is divided into Q equal intervals covered by $Q-1$ band-pass filters ψ_λ of bandwidth $2\pi/T$. Note that the lower part of the filter bank is equivalent to a STFT and that this abrupt change from Q-constant to STFT is sometimes replaced by a smooth transition [187]. Sometimes, in fact, despite not being perfectly Q-constant, a smooth transition is more desirable than a constant bandwidth at lower frequencies. Finally, the filter bank is completed by adding one more low-pass filter $\phi(t)$ with bandwidth $2\pi/T$ and $|\hat{\phi}(\omega)| \leq 1$. Therefore, the *wavelet transform* operator Wx can be modified to include also the low-frequency (non-wavelet) function $\phi(t)$ [79] :

$$Wx = \left\{x \star \phi(t),\ x \star \psi_\lambda(t)\right\}_{t\in\mathbb{R},\lambda\in\Lambda} \quad , \tag{3.49}$$

where Λ indicates the set of indexes per each central frequency ω_λ of the functions ψ_λ. The operator Wx is contractive, that is:

$$\|Wx\|_2^2 \leq \|x\|_2^2. \tag{3.50}$$

In fact, because of the construction described above, the following relation [82] holds for the sum of the squared modulus[6]:

[6] In general, the relation (3.51) is valid accounting for at most a multiplicative factor, even when the bandwidth of the filters is not defined as ω_λ/Q but using different criteria.

$$A(\omega) = |\hat{\phi}(\omega)|^2 + \frac{1}{2}\sum_{\lambda\in\Lambda}\left(|\hat{\psi}_\lambda(\omega)|^2 + |\hat{\psi}_\lambda(-\omega)|^2\right) \leq 1 \quad \forall\omega\in\mathbb{R}. \tag{3.51}$$

Multiplying the previous relation by $|\hat{x}(\omega)|^2$, integrating over ω, and applying Plancherel's formula returns (3.50). Note that $A(\omega)$ is also > 0 and, with analogous steps:

$$(1-\alpha)\|x\|_2^2 \leq \|Wx\|_2^2 \qquad 0 \leq \alpha < 1, \tag{3.52}$$

which implies that the function $x(t)$ can be recovered from Wx as its stable inverse [80, pp. 168–170].

When computing the mel coefficients $|x\star\psi_\lambda|^2\star|\phi|^2(t)$ in (3.31), the squares do not play any important role in the effectiveness of the representation. Moreover, they make the functions more difficult to handle and introduce a non-linearity that might amplify undesired outliers. Hence, it is possible to consider $|x\star\psi_\lambda|\star|\phi|(t) = |x\star\psi_\lambda|\star\phi(t)$ instead of the original relation in (3.31) without any loss of generality. Recalling the interpretation in terms of analytic wavelets, note that calculating the modulus of $x\star\psi_\lambda(t)$ is a necessary operation to calculate the average obtained by the convolution with $\phi(t)$. Indeed, without introducing a non-linearity, averaging the wavelet coefficients $x\star\psi_\lambda$ would return zero. Besides, the modulus does not alter the contractive property of Wx (indeed $|x-y| \geq ||x|-|y||$), preserves the energy, and because of the redundancy of the wavelet representation, does not cause any loss of information as it would be for the Fourier transform [188].

3.2.3.4 Wavelets scattering transform

It has been shown that $x\star|\phi|(t)$ provides translation invariant coefficients while $|x\star\psi_\lambda|(t)$ provides coefficients that are robust against deformations without losing information. The operator *wavelet modulus transform* is defined as:

$$|W|x = \left\{x\star\phi(t), |x\star\psi_\lambda(t)|\right\}_{t\in\mathbb{R},\lambda\in\Lambda}. \tag{3.53}$$

As for Wx, $|W|x$ is invertible [188] and contractive. The contractive property is important to guarantee that the representation provides a distance that is no bigger than the actual distance between two different signals, that is:

$$\| \, |W|x - |W|y \, \|_2^2 \leq \|Wx - Wy\|_2^2 \leq \|x - y\|_2^2. \tag{3.54}$$

This property keeps the representation stable if the operator is applied multiple times and limits the effect of additive noise.

It is here remarked that, since ψ_λ is assumed analytic, $|x \star \psi_\lambda|(t)$ calculates the modulus of the analytic function $x \star \psi_\lambda(t)$. This operation extracts the envelope of the latter and reshapes its spectrum. Since the envelope is smoother, the reshaped spectrum tends to translate the energy of the original analytic signal toward the low frequencies. Coefficients $|x \star \psi_\lambda|(t)$ are simply equivariant since the representation shifts with the translation of the input. This is an issue when time invariance is required for the analysis. For MFCC, the problem is tackled by averaging, that is, by calculating the convolution with $\phi(t)$. However, this operation causes a loss of information since it keeps only the very low-frequency components of the envelope $|x \star \psi_\lambda|(t)$. The rest is lost. The *wavelet scattering transform* provides a solution to this issue with a multilayer iterative architecture.

In layer zero, the information lost by the invariant features $S_0 x(t) = x \star \phi(t)$ are recovered by calculating the wavelet modulus transform:

$$|W_1|x = \left\{ x \star \phi(t), \; |x \star \psi_{\lambda_1}(t)| \right\}_{t \in \mathbb{R}, \lambda_1 \in \Lambda_1}. \tag{3.55}$$

Here Λ_1 refers to the set of filters used in layer one. To make $|x \star \psi_{\lambda_1}(t)|$ time invariant, a convolution with $\phi(t)$ is calculated. This operation returns the features for the layer one $S_1 x(t, \lambda_1) = |x \star \psi_{\lambda_1}| \star \phi(t)$ for each $\lambda_1 \in \Lambda_1$, which are time invariant and stable against time-warping deformation. As for $S_0 x(t)$, however, the low pass filter discards most of the information associated with the envelopes $|x \star \psi_{\lambda_1}|$. Again, this information is maintained by calculating the wavelet modulus

transform for layer two per each $\lambda_1 \in \Lambda_1$:

$$|W_2||x \star \psi_{\lambda_1}| \\ = \left\{ |x \star \psi_{\lambda_1}| \star \phi(t),\ ||x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}(t)| \right\}_{t \in \mathbb{R}, \lambda_2 \in \Lambda_2} \ \forall \lambda_1 \in \Lambda_1. \tag{3.56}$$

The quantity $||x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}(t)|$ computes a new set of envelopes that can be averaged by the low pass filter $\phi(t)$ in order to provide invariant and stable features for layer two $S_2 x(t, \lambda_1, \lambda_2) = ||x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}| \star \phi(t)$ for each $\lambda_1 \in \Lambda_1$ and $\lambda_2 \in \Lambda_2$.

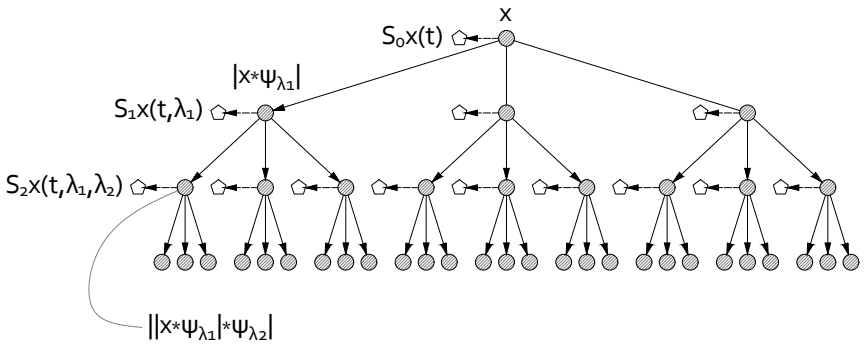


Figure 3.8: *Topology of the scattering transform network. At each layer, the invariant coefficients S_m are calculated by filtering with the low pass filter $\phi(t)$. Information lost to achieve invariance is recovered in the subsequent layer by calculating a new wavelet modulus transform. The root of the tree is given by the input signal $x(t)$.*

This procedure can be repeated for the higher order layers to yield the invariants and to keep the information discarded while achieving invariance. Figure 3.8 illustrates the architecture of the network. The scattering coefficients for the layer m of the *wavelet scattering transform (WST)* are indicated as:

$$S_m x(t, \lambda_1, ...\lambda_m) = |||x \star \psi_{\lambda_1}| \star ...| \star \psi_{\lambda_m}| \star \phi(t), \tag{3.57}$$

where $S_m x$ denotes the *wavelet scattering operator* for layer m, and $\lambda_1, ..., \lambda_m$ indicate a specific *path* of the scattering network. Note that, since the wavelet modulus transform is contractive, the scattering transform is contractive too, and it is stable to additive noise (a small perturbation affects the representation only marginally). Besides, since the envelopes at each step shift the energy toward

lower frequencies, the original signal energy is progressively captured by low-pass filters, and coefficients tend to decrease when a new layer is calculated. Usually, for audio signals, the coefficients beyond layers three or four are negligible. This number, however, is also related to the length T of the window. As T increases, the band of the low-pass filter ϕ becomes narrower. This means that a greater portion of the energy is scattered to the next layer. For short windows, e.g. $20ms$, layer one might be sufficient to capture most of the energy, making MFCC performance comparable to scattering transform. However, a larger window is required to capture large sound structures, and MFCC loses the information that the scattering transform represents on higher-order layers. For audio signals, a good starting point for the resolution of the filter is $Q_1 = 8$ and $Q_n = 1$ for $n > 1$ [79]. However, this choice depends on the application, and other values might be required for better performance.

Features extracted with wavelets scattering transform can be optimised by performing a normalisation between coefficients of adjacent layers. In its general form, the normalisation can be written as:

$$\bar{S}_m x(t, \lambda_1, ..., \lambda_m) = \frac{S_m x(t, \lambda_1, ..., \lambda_m)}{S_{m-1} x(t, \lambda_1, ..., \lambda_{m-1}) + \varepsilon}, \tag{3.58}$$

where ε is a silence threshold to set $\bar{S}_m x = 0$ if $x = 0$. For instance, for $m = 1$, the normalisation makes the representation invariant when $x(t)$ is multiplied by a constant. For all the orders, normalisation generally decorrelates coefficients belonging to different layers. Further implementation details for discrete signals and other observations to reduce the amount of useful coefficient required are reported in chapter 6.

Wavelet scattering networks exhibit similarities with convolutional networks [189] in terms of both topology and performed operations. However, the most important difference is related to the fact that scattering networks do not need to be trained since filters and pooling operations are already given. This provides better performance when training data are scarce, but its predefined structure might cap the performance in the case of complex sound structures. Moreover, scattering

features are extracted at each layer, while convolutional networks usually extract features from the last layer.

3.3 Feature transformation and reduction

Dealing with large datasets requires a lot of resources in terms of hardware to store and process the data. Besides, processing raw data directly is not a good practice for several reasons. For instance, the source representation might be unnecessarily complex and even too sensitive to undesired inputs such as additive noise. When compression is applied to sounds or images, the size of the primary data is remarkably reduced with negligible loss of quality. This transformation is achieved through algorithms that provide a more meaningful and compact representation of the data while discarding unnecessary redundancy. For instance, the standard *jpeg2000* [190] uses wavelet representation to describe only those portions of the images that carry interesting information (e.g. the edge of an object). The audio standard *MP3*[7] achieves the same result via discrete cosine transform [191]. Similarly, representing the features in a lower-dimensional space allows a reduction of the resources necessary for storage and computation and, in machine learning, usually yields better results. In the previous section, it has been shown how a signal can be represented to provide robust and invariant features. The next step is understanding how to transform/reduce these features to yield something more convenient for machine learning tasks. Obviously, any good transformation shall not lose any relevant information.

3.3.1 Principal component analysis

Principal component analysis (PCA) is probably one of the most widespread techniques for feature transformation/reduction. It has been implemented in many variants [192],[193],[194] and the underlying idea is to transform the data to obtain a new set of equivalent and decorrelated variables that can be sorted in relation to their variance. Although the transformed variables often lose trivial connections

[7] To the best of our knowledge, at the time of writing this text, no audio compression standard using wavelets has been published.

with meaningful physical quantities, the transformed representation generally allows better visualisation, dimensionality reduction, and improved clustering. In the next section, it is shown the case where the transformation between the old and new set of coordinates is linear. In section 3.3.1.2, the same concept is extended to non-linear transformations.

3.3.1.1 Linear PCA

Linear PCA [195] is usually implemented either as a solution to an eigenvalues/eigenvectors problem or from singular values decomposition. Under certain conditions, however, these two solutions provide equivalent results. Here, the first option is described because it allows the extension to the non-linear case.

Data is assumed organised in a matrix $\boldsymbol{X} \in \mathbb{R}^{N \times D}$, with N examples and D variables. The examples are given by the row vectors $\boldsymbol{x}_{n,:} = \boldsymbol{X}_{(n,:)} \in \mathbb{R}^{1 \times D}\ \forall\ n \in [1:N]$, while the variables by the column vectors $\boldsymbol{x}_{:,d} = \boldsymbol{X}_{(:,d)} \in \mathbb{R}^{N \times 1}\ \forall\ d \in [1:D]$. For audio, this can be obtained by slicing signal representations in N arrays of D samples. Firstly, it is assumed that each variable $\boldsymbol{x}_{:,d}$, is centred. This condition can be obtained by shifting to zero the mean of each column. Issues related to non-centred variables are accounted for later. The aim of the principal component analysis is to find a linear transformation $\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{W} \in \mathbb{R}^{N \times L}$ where the new variables $\boldsymbol{z}_{:,l} = \boldsymbol{Z}_{(:,l)} \in \mathbb{R}^{N \times 1}$, the *principal components* (PCs), exhibit maximised and descending variance, from the first PC $\boldsymbol{z}_{:,1}$ to the last $\boldsymbol{z}_{:,L}$. The new reference system is given by the *loadings* of the matrix $\boldsymbol{W} \in \mathbb{R}^{D \times L}$, that is the L orthonormal vectors $\boldsymbol{w}_{:,l} = \boldsymbol{W}_{(:,l)} \in \mathbb{R}^{D \times 1}$. The loadings represent the direction of the PCs in the original reference system. The vectors $\boldsymbol{z}_{n,:} = \boldsymbol{Z}_{(n,:)} \in \mathbb{R}^{1 \times L}$ are called *scores* since they represent each example in the new reference system. The scores of an example are given by a numerical value per each PC.

The principal components can be found one by one, starting from the first. To find the first PC, it is necessary to determine the first vector $\boldsymbol{w}_{:,1}$ as:

$$\boldsymbol{w}_{:,1} = \underset{\boldsymbol{w}_{:,1}\big|_{\|\boldsymbol{w}_{:,1}\|_2^2=1}}{\arg\max} \sum_{k=1}^{N} z_{k,1}^2 = \underset{\boldsymbol{w}_{:,1}\big|_{\|\boldsymbol{w}_{:,1}\|_2^2=1}}{\arg\max}\ \boldsymbol{w}_{:,1}^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{w}_{:,1}\ , \tag{3.59}$$

where ^{T} denotes the transpose. The maximisation problem above with the condition $\|\boldsymbol{w}_{:,1}\|_2^2 = 1$ is equivalent to maximising the Lagrangian function:

$$\mathscr{L}(\boldsymbol{w}_{:,1}, \lambda) = \boldsymbol{w}_{:,1}^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{w}_{:,1} - \lambda(\boldsymbol{w}_{:,1}^T \boldsymbol{w}_{:,1} - 1) \ , \tag{3.60}$$

with λ being the Lagrange multiplier. The solution can be obtained by finding the vector $\boldsymbol{w}_{:,1}$ that makes the derivative null, that is:

$$\boldsymbol{X}^T \boldsymbol{X} \boldsymbol{w}_{:,1} = \lambda \boldsymbol{w}_{:,1} \ . \tag{3.61}$$

Hence, recalling the definitions of eigenvalue and eigenvector, the variance of $\boldsymbol{z}_{:,1}$ is maximised by one of the eigenvectors $\bar{\boldsymbol{w}}$ of $\boldsymbol{X}^T \boldsymbol{X}$ corresponding to one of the eigenvalues $\bar{\lambda}$. Since $\bar{\boldsymbol{w}}^T \boldsymbol{X}^T \boldsymbol{X} \bar{\boldsymbol{w}} = \bar{\lambda} \bar{\boldsymbol{w}}^T \bar{\boldsymbol{w}} = \bar{\lambda}$, the eigenvector $\bar{\boldsymbol{w}}$ that maximises the variance is the one corresponding to the highest eigenvalue. Equation (3.61) does not change by multiplying the eigenvector by -1, meaning that its direction is irrelevant. Besides, note that the matrix $\boldsymbol{X}^T \boldsymbol{X} \in \mathbb{R}^{D \times D}$ is real, symmetric and positive semi-definite, [8] [9] so its D eigenvalues are all real and ≥ 0. Furthermore, the associated D eigenvectors are all orthogonal. To find the second principal component, the matrix $\boldsymbol{X}$ can be decomposed as:

$$\boldsymbol{X} = \tilde{\boldsymbol{X}} + \boldsymbol{X} \boldsymbol{w}_{:,1} \boldsymbol{w}_{:,1}^T \ , \tag{3.64}$$

where $\tilde{\boldsymbol{X}}$ is obtained from $\boldsymbol{X}$ by removing the variations in the direction of the first principal component. Recalling that the direction of the second principal component must be orthogonal to the first, that is $\boldsymbol{w}_{:,1}^T \boldsymbol{w}_{:,2} = 0$, and repeating

[8] A matrix $\boldsymbol{A} \in \mathbb{R}^{D \times D}$ is positive semi-definite (definite) if, for any non-null vector $\boldsymbol{y} \in \mathbb{R}^{D \times 1}$, it returns:

$$\boldsymbol{y}^T \boldsymbol{A} \boldsymbol{y} \geq 0 \quad (>0) \tag{3.62}$$

For positive semi-definite (definite) matrices, the eigenvalues are always ≥ 0 (>0) since, if $\boldsymbol{y}$ is an eigenvector and λ its eigenvalue,

$$\boldsymbol{y}^T \lambda \boldsymbol{y} = \boldsymbol{y}^T \boldsymbol{A} \boldsymbol{y} \geq 0 \quad (>0), \tag{3.63}$$

where $\boldsymbol{y}^T \lambda \boldsymbol{y}$ has the same sign of λ.

the steps above for $\boldsymbol{w}_{:,2}$, the highest eigenvalue of $\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{X}}$ is the second highest eigenvalue of $\boldsymbol{X}^T\boldsymbol{X}$. Hence, $\boldsymbol{w}_{:,2}$ is the eigenvector associated with the second highest eigenvalue. The procedure above can be repeated for all the L highest eigenvalues (certainly even when $L = D$) by removing from $\tilde{\boldsymbol{X}}$ the variation in the direction of all the previous principal components.

Usually, to calculate the principal components, the mean of the variables $\boldsymbol{x}_{:,d}$ is assumed to be null. Through the matrix $\boldsymbol{W}$, the principal components analysis provides a rotation of the original reference system to align the first principal component to the direction of the maximum variation.

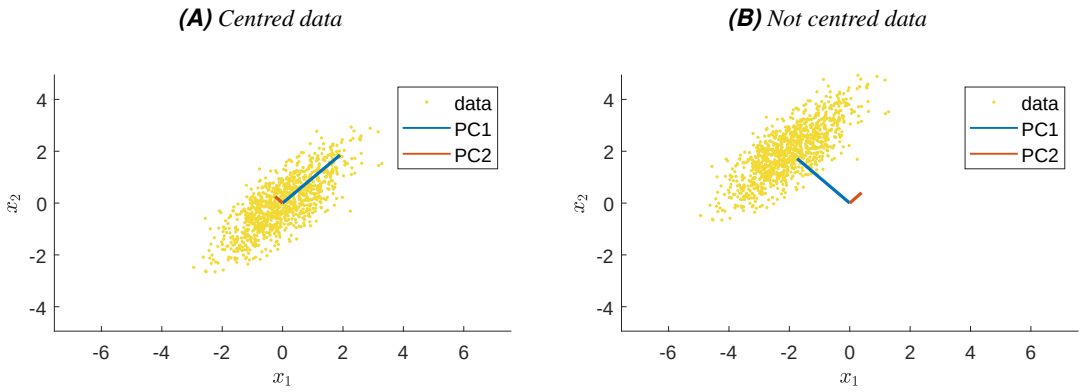


Figure 3.9: *PCA for centred and not centred data. The yellow points indicate a random variable obtained as a realisation of a Gaussian process. The blue line indicates the direction of the eigenvector associated with PC1, while the red one is associated with PC2. Both lines have been scaled proportionally to the related eigenvalues. For centred data (left), PC1 correctly indicates the direction of the maximum variance. When data are not centred, PC1 indicates the direction of the cluster, missing the correct max variance direction.*

If data is not centred but clustered away from the origin, the first principal component is aligned to the direction of the centre of the cluster, which is, in general, different from the direction of the maximum variation. Moreover, if the cluster's distance from the origin is much bigger than the size of the cluster, the variance is incorrectly dominated by the cluster's distance, and it is mainly associated only with the first principal component. In this case, PCA does not provide useful information about the variance of the variables (figure 3.9). This

issue is usually tackled by calculating the PCA using the *sample covariance matrix* $\boldsymbol{K}_{xx}$ [9] instead of $\boldsymbol{X}^T\boldsymbol{X}$. Indeed, $\boldsymbol{K}_{xx}$ centers the variables by definition, and it simply becomes $\frac{1}{N-1}\boldsymbol{X}^T\boldsymbol{X}$ when data are already centred. Note that the sample covariance matrix of $\boldsymbol{Z}$ is not null only along the main diagonal. This is usually recalled by saying that the PCA diagonalises the covariance matrix. Indeed, $\boldsymbol{z}_{:,k}^T\boldsymbol{z}_{:,h} = \boldsymbol{w}_{:,k}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}_{:,h} = \lambda_h\boldsymbol{w}_{:,k}^T\boldsymbol{w}_{:,h}$, which is not null only for $h = k$.

An interesting property of PCA [197, p. 388] is that it minimises the reconstruction error in the sense of the squared norm $\|\boldsymbol{X} - \boldsymbol{Z}_L\boldsymbol{W}_L^T\|_F^2$ [10], where the subscript L indicates that PCA is obtained by accounting only for the first L principal components. Hence, by choosing $L < D$ it is possible to reduce the dimensionality, meaning that each example $\boldsymbol{x}_{n,:}$ can be represented by L variables instead of D while keeping the best approximation of $\boldsymbol{x}_{n,:}$ in the sense of the squared norm.

[9] Given the sample data matrix $\boldsymbol{X} \in \mathbb{R}^{N\times D}$ with N examples of D variables:

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1D} \\ x_{21} & x_{22} & \dots & x_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \dots & x_{ND} \end{bmatrix}, \tag{3.65}$$

the *sample covariance matrix* $\boldsymbol{K}_{xx}$ is the symmetric matrix defined as [196, p. 177]:

$$\boldsymbol{K}_{xx} = \frac{1}{N-1}\begin{bmatrix} k_{11} & k_{12} & \dots & k_{1D} \\ k_{21} & k_{22} & \dots & k_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ k_{D1} & k_{D2} & \dots & k_{DD} \end{bmatrix}, \tag{3.66}$$

where

$$k_{jj} = \sum_{i=1}^{N}(x_{ij} - \bar{x}_j)^2 \qquad \forall\ j = [1:D] \tag{3.67}$$

$$k_{hj} = k_{jh} = \sum_{i=1}^{N}(x_{ij} - \bar{x}_j)(x_{ih} - \bar{x}_h) \qquad 1 \le h, j \le D \quad j \ne h \tag{3.68}$$

$$\bar{x}_j = \frac{1}{N}\sum_{i=1}^{N} x_{i,j}. \tag{3.69}$$

Note that $\boldsymbol{K}_{xx}$ can be written as:

$$\boldsymbol{K}_{xx} = \frac{1}{N-1}\sum_{i=1}^{N}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T \quad with \quad \bar{\boldsymbol{x}} = \frac{1}{N}\sum_{k=1}^{N}\boldsymbol{X}_{(k,:)}^T \quad \boldsymbol{x}_i = \boldsymbol{X}_{(i,:)}^T \tag{3.70}$$

For centered data $\bar{x}_{:,j} = 0\ \ \forall\ j = [1:D]$ and $\boldsymbol{K}_{xx} = \frac{1}{N-1}\boldsymbol{X}^T\boldsymbol{X}$. $\boldsymbol{K}_{xx}$ is positive semi-definite since per each vector $\boldsymbol{y} \in \mathbb{R}^{D\times 1}$:

$$\boldsymbol{y}^T\boldsymbol{K}_{xx}\boldsymbol{y} = \frac{1}{N-1}\boldsymbol{y}^T\Big[\sum_{i=1}^{N}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T\Big]\boldsymbol{y} = \frac{1}{N-1}\sum_{i=1}^{N}\big[(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T\boldsymbol{y}\big]^2 \ge 0 \tag{3.71}$$

Since the variance and the eigenvalue of each principal component are directly related, it is possible to define a threshold in terms of the percentage of the total sum of the eigenvalues and to keep only the associated eigenvectors. Commonly, most of the variance is associated with the first few principal components. If the number of components is reduced to two or three, it is possible to represent data defined in a higher dimensional space using a simple 2D or 3D plot.

3.3.1.2 Kernel PCA

PCA performs a linear transformation between two different reference systems. Depending on the data, however, this linear transformation may not be able to enhance the representation and provide relevant benefits to the other machine-learning tasks. Sometimes, a better solution comes from assuming the source data given in a space S as a simplified representation of a different one living in a much higher dimensional space S'. Classes that appear strongly tangled in S, after performing PCA in S' might become better clustered and linearly separable (figure 3.10). The optimal space S', however, is generally unknown in advance. Besides, since the dimension of S' is usually much larger or infinite, even assuming a certain known map function $\boldsymbol{\xi} : S \rightarrow S'$, calculating PCA in S' might be unfeasible.

An elegant solution comes from the *Kernel method* for PCA [198], which, under certain conditions for $\boldsymbol{\xi}$ and by using the so-called *kernel trick*, allows the calculation of PCA in S' without actually performing any calculation in the higher dimensional space. With the same notations used in the previous paragraph, it is here assumed that the map function $\boldsymbol{\xi}(\boldsymbol{x}_{i,:})$ maps any example $\boldsymbol{x}_{i,:}$ from $S \in \mathbb{R}^D$ to a different space $S' \in \mathbb{R}^H$, with $H > D$. $\boldsymbol{\Xi}$ indicates the matrix of size $N \times H$ that collects the representation of the examples in S'. Firstly, it is assumed that data mapped in S' are centred. This condition is generally not verified, but this issue is accounted for toward the end of this section.

As from the previous section, performing PCA in S' would require calculating

[10] The Frobenius norm of a matrix is defined as:

$$\|\boldsymbol{A}\|_F^2 = \sum_{i=1}^{N}\sum_{j=1}^{M} a_{ij}^2 = trace(\boldsymbol{A}^T\boldsymbol{A}) \tag{3.72}$$

eigenvalues and eigenvectors of $\Xi^T\Xi$. However, since $\Xi^T\Xi$ is of size $H \times H$, with H potentially infinite, the above calculation is generally not feasible. Instead, let $\boldsymbol{U}$ be a matrix containing the normalised eigenvectors of $\Xi\Xi^T$ arranged by columns and $\boldsymbol{\Lambda}$ a matrix with the corresponding eigenvalues along the principal diagonal. With these assumptions, $\Xi\Xi^T\boldsymbol{U} = \boldsymbol{U}\boldsymbol{\Lambda}$. Multiplying by Ξ^T yields:

$$\Xi^T\Xi\Xi^T\boldsymbol{U} = \Xi^T\boldsymbol{U}\boldsymbol{\Lambda}, \tag{3.73}$$

which means that $\Xi^T\boldsymbol{U}$ are the eigenvectors of $\Xi^T\Xi$ with the same eigenvalues in $\boldsymbol{\Lambda}$. Note that $\Xi^T\Xi$ has the same number of independent eigenvectors of $\Xi\Xi^T$ so $\Xi^T\boldsymbol{U}$ includes them all[11]. As for the PCA, eigenvectors of $\Xi^T\Xi$ shall exhibit unitary norm and, since $\boldsymbol{u}_{:,j}^T\Xi\Xi^T\boldsymbol{u}_{:,j} = \boldsymbol{u}_{:,j}^T\boldsymbol{u}_{:,j}\lambda_j = \|\boldsymbol{u}_{:,j}\|_2^2\lambda_j = \lambda_j$, the normalised eigenvectors will be $\Xi^T\boldsymbol{U}\boldsymbol{\Lambda}^{-\frac{1}{2}}$.

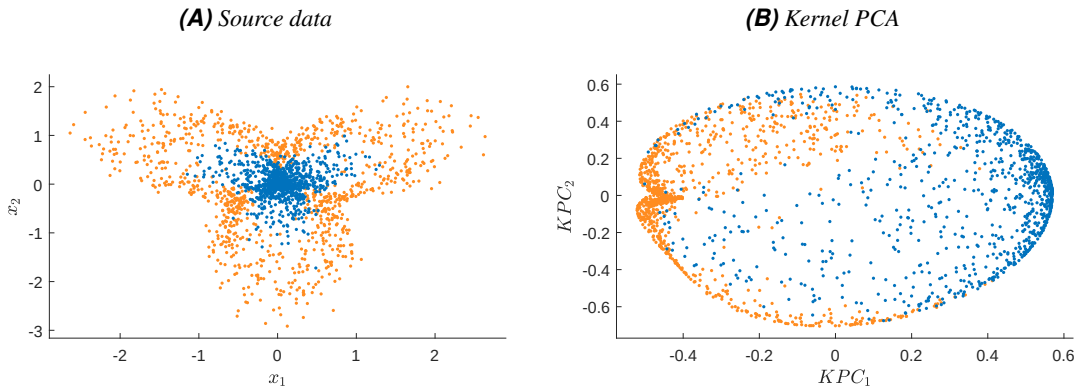


Figure 3.10: *Effect of the non-linearity on PCA. Picture A shows a set of data $\in \mathbb{R}^2$ that are not linearly separable. After performing kernel PCA (picture B), the two classes are better clustered and can be linearly separated with just a few outliers. Good separation can be achieved with just one principal component. The nonlinearity is given by a Gaussian Kernel.*

As seen in the previous section where $\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{W}$, the principal components $\boldsymbol{Z}'$ in the transformed space S' can be found by projecting the transformed data Ξ on the eigenvectors found above, that is:

$$\boldsymbol{Z}' = \Xi\Xi^T\boldsymbol{U}\boldsymbol{\Lambda}^{-\frac{1}{2}}. \tag{3.74}$$

[11] Given matrix $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ with rank $k \leq \min(M,N)$, the symmetric matrices $\boldsymbol{A}\boldsymbol{A}^T$ and $\boldsymbol{A}^T\boldsymbol{A}$ have rank k and k linearly independent eigenvectors.

In the previous equation, $\boldsymbol{U}$ and $\boldsymbol{\Lambda}$ can be calculated but $\boldsymbol{\Xi}$ is unknown. However, $\boldsymbol{\Xi}$ appears only in the form $\boldsymbol{\Xi}\boldsymbol{\Xi}^T$, which is a matrix of size $N \times N$, and it can be calculated under the conditions reported below.

The Mercer's theorem [199] guarantees that, given a *kernel function* $\kappa : S^2 \to \mathbb{R}$ defined to be:

$$\begin{aligned} &\kappa(\boldsymbol{x}_{h,:}\ ,\ \boldsymbol{x}_{j,:}) \geq 0 \\ &\qquad\qquad\qquad\qquad\qquad\qquad \forall\ h, j \in [1:N] \\ &\kappa(\boldsymbol{x}_{h,:}\ ,\ \boldsymbol{x}_{j,:}) = \kappa(\boldsymbol{x}_{j,:}\ ,\ \boldsymbol{x}_{h,:}), \end{aligned} \tag{3.75}$$

if the related *Gram matrix*:

$$\boldsymbol{K} = \begin{bmatrix} \kappa(\boldsymbol{x}_{1,:}\ ,\ \boldsymbol{x}_{1,:}) & \dots & \kappa(\boldsymbol{x}_{1,:}\ ,\ \boldsymbol{x}_{N,:}) \\ \vdots & \ddots & \vdots \\ \kappa(\boldsymbol{x}_{1,:}\ ,\ \boldsymbol{x}_{1,:}) & \dots & \kappa(\boldsymbol{x}_{1,:}\ ,\ \boldsymbol{x}_{N,:}) \end{bmatrix} \tag{3.76}$$

is positive definite, then there exists a map function $\boldsymbol{\xi}(\boldsymbol{x})$ so that the kernel function computes its inner product. Hence, if the non-linear transformation is chosen by choosing first its associated Mercer's kernel, $\boldsymbol{\Xi}\boldsymbol{\Xi}^T$ can be written as:

$$\begin{aligned} \boldsymbol{K} &= \boldsymbol{\Xi}\boldsymbol{\Xi}^T \\ &= \begin{bmatrix} \boldsymbol{\xi}(\boldsymbol{x}_{1,:})\boldsymbol{\xi}^T(\boldsymbol{x}_{1,:}) & \dots & \boldsymbol{\xi}(\boldsymbol{x}_{1,:})\boldsymbol{\xi}^T(\boldsymbol{x}_{N,:}) \\ \vdots & \ddots & \vdots \\ \boldsymbol{\xi}(\boldsymbol{x}_{N,:})\boldsymbol{\xi}^T(\boldsymbol{x}_{1,:}) & \dots & \boldsymbol{\xi}(\boldsymbol{x}_{N,:})\boldsymbol{\xi}^T(\boldsymbol{x}_{N,:}) \end{bmatrix} = \begin{bmatrix} \kappa(\boldsymbol{x}_{1,:}\ ,\ \boldsymbol{x}_{1,:}) & \dots & \kappa(\boldsymbol{x}_{1,:}\ ,\ \boldsymbol{x}_{N,:}) \\ \vdots & \ddots & \vdots \\ \kappa(\boldsymbol{x}_{1,:}\ ,\ \boldsymbol{x}_{1,:}) & \dots & \kappa(\boldsymbol{x}_{1,:}\ ,\ \boldsymbol{x}_{N,:}) \end{bmatrix} \end{aligned} \tag{3.77}$$

and this allows the calculation of the PCA in the space S'. Replacing the inner product with a call to the kernel function is usually referred to as the *kernel trick* and is the crucial point of kernel methods [197, p. 488].

As seen for PCA, to make the transformation useful, data should be centred. However, in this case, centring cannot be done directly since no explicit representation is given in S'. To solve the problem, once the kernel κ has been chosen, indi-

cating the centred version of the vector $\boldsymbol{\xi}(\boldsymbol{x}_{h,:})$ as $\bar{\boldsymbol{\xi}}_{h,:} = \boldsymbol{\xi}(\boldsymbol{x}_{h,:}) - \frac{1}{N}\sum_{j=1}^{N}\boldsymbol{\xi}(\boldsymbol{x}_{j,:})$, the items of the centred Gram Matrix can be calculated using the inner product and the kernel trick one more time:

$$\begin{aligned}\bar{K}_{h,k} &= \bar{\boldsymbol{\xi}}_{h,:}^{T}\bar{\boldsymbol{\xi}}_{k,:} \\ &= \boldsymbol{\xi}_{h,:}^{T}\boldsymbol{\xi}_{k,:} - \frac{1}{N}\sum_{j=1}^{N}\boldsymbol{\xi}_{h,:}^{T}\boldsymbol{\xi}_{j,:} - \frac{1}{N}\sum_{j=1}^{N}\boldsymbol{\xi}_{k,:}^{T}\boldsymbol{\xi}_{j,:} + \frac{1}{N^2}\sum_{j=1}^{N}\sum_{l=1}^{N}\boldsymbol{\xi}_{j,:}^{T}\boldsymbol{\xi}_{l,:} \\ &= \kappa(\boldsymbol{x}_{h,:}, \boldsymbol{x}_{k,:}) - \frac{1}{N}\sum_{j=1}^{N}\kappa(\boldsymbol{x}_{h,:}, \boldsymbol{x}_{j,:}) - \frac{1}{N}\sum_{j=1}^{N}\kappa(\boldsymbol{x}_{k,:}, \boldsymbol{x}_{j,:}) + \frac{1}{N^2}\sum_{j=1}^{N}\sum_{l=1}^{N}\kappa(\boldsymbol{x}_{j,:}, \boldsymbol{x}_{l,:}).\end{aligned} \tag{3.78}$$

The previous equation can be written in matrix form as:

$$\bar{\boldsymbol{K}} = \bar{\boldsymbol{\Xi}}\bar{\boldsymbol{\Xi}}^{T} = \boldsymbol{K} - \frac{1}{N}\mathbf{1}_N\boldsymbol{K} - \frac{1}{N}\boldsymbol{K}\mathbf{1}_N + \frac{1}{N^2}\mathbf{1}_N\boldsymbol{K}\mathbf{1}_N = \boldsymbol{HKH}, \tag{3.79}$$

where $\boldsymbol{H} = \boldsymbol{I} - \frac{1}{N}\mathbf{1}_N$, $\boldsymbol{I}$ is the identity matrix of size $N \times N$, and $\mathbf{1}_N$ is the all-one squared matrix of size $N \times N$. Note that eigenvectors $\boldsymbol{U}$ and eigenvalues $\boldsymbol{\Lambda}$ in (3.73) need to be calculated using $\bar{\boldsymbol{K}} = \bar{\boldsymbol{\Xi}}\bar{\boldsymbol{\Xi}}^{T}$ instead of $\boldsymbol{K} = \boldsymbol{\Xi}\boldsymbol{\Xi}^{T}$.

When using kernel PCA, the main issue is finding a transformation which can be expressed in terms of its kernel and is appropriate for the application. In general, it is not easy to prove that a kernel is Mercer's [197, p.], but Mercer's kernels can be combined to obtain new Mercer's kernels. For instance, this approach has been used in *multiple kernel learning* [200] to find sub-optimal kernels. Some examples are given in chapter 6.

3.4 Classification and event detection

Once a signal has been represented with some set of features, transformed and reduced, the last step consists in detecting and classifying possible acoustic events happening along the audio recording. This last step provides information that is meaningful to humans. Detecting an event means identifying its start and its end time in the recording. This information is complemented with the classification, which, in this work, means assigning a descriptive audio label taken from

a set of given audio classes. The software entity that performs this latter operation is called *classifier*. In the *discriminative* approach (figure 3.11), a classifier tries to predict the labels by describing some boundaries that can discriminate between different classes. No further modelling assumptions are required. For instance, K-nearest neighbours (KNN) (section 3.4.1), support vector machines (SVM) (section 3.4.2), and *random forests* [201] are classification techniques that can be defined as discriminative. Accounting for the differences, traditional *neural networks* can also be considered discriminative.

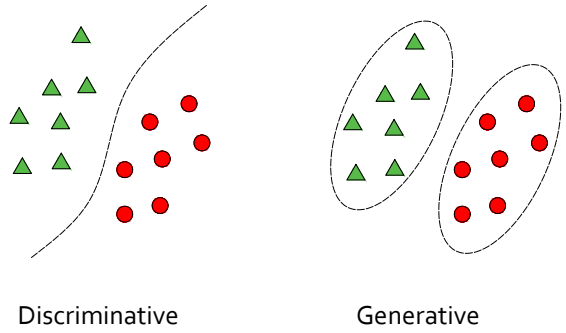


Figure 3.11: *Discriminative versus generative classifiers. In the discriminative approach, classes are separated by boundaries in the representation space. The generative approach tries to model the underlying process that generates the data.*

On the other hand, a *generative* approach (figure 3.11) tries to work out a model that is intended to describe the underlying process that generates the data. They are called generative in the sense that it is possible, if needed, to generate new data points from the model of the process. Some examples of generative techniques are *naive-Bayes* classifiers [202, p. 258], *Gaussian mixture models* (GMM) [38, p. 115] and *Hidden Markov Models* (HMM) [203]. Although it is not possible to provide an always-true rule, a discriminative approach is arguably more indicated for this work. This assumption is motivated by the fact that noise-like events might be very difficult to model, and outliers are better managed by defining simple separation boundaries between classes [38, p. 27]. Hence, the following sections are mainly focused on discriminative approaches.

Classifiers exploit reference examples to assign meaning to similarities between different groups of features. From this point of view, classification is always a

supervised learning operation. Nevertheless, machine learning algorithms can take advantage of operations where data are processed independently from their labels to extract underlying relevant information. PCA, for instance, does not require any labels to be performed and can be considered an *unsupervised* operation. A supervised approach can also be adopted at the feature extraction level to provide a more discriminative representation.

Given the number of solutions reported in the literature, it is not possible to provide an exhaustive description of the field, and the following sections describe only those techniques that will be recalled later in chapter 6.

3.4.1 K-nearest neighbours

The *K-nearest neighbours (KNN)* algorithm [204], in its naive implementation, provides one of the simplest forms of classifiers. Given its simplicity, it is often considered a benchmark for more complex techniques. The basic idea is to measure the distance between a test example and a set of references to select the nearest K of them (figure 3.12). The test example is classified by counting the labels of the nearest K references and assigning the label of the most represented class among the neighbours (*the majority vote*). In the case of $K = 1$ the assigned label corresponds to the label of the nearest reference. If $(\boldsymbol{x}, y) \in \mathbb{R}^D \times C$ is a test example, and $\{(\boldsymbol{r}_i, c_i)\}_{i=1}^N \in \mathbb{R}^D \times C$ is a set of N reference examples, the distance between $(\boldsymbol{x}, y)$ and $(\boldsymbol{r}_i, c_i)$ can be written as:

$$d_{\boldsymbol{x},\boldsymbol{r}_i} = \|\boldsymbol{x} - \boldsymbol{r}_i\|, \tag{3.80}$$

where $\|\cdot\|$ is some kind of metric, C the set of reference classes, and D the dimension of the features. By sorting the distances d_{x,r_i} and selecting the smallest K, it is possible to formulate a prediction for the class y. Choosing the optimal number of neighbours K generally reduces the sensitivity to noisy examples and improves the classifier performance. This is usually done with some heuristic supervised approach [205].

In its basic implementation, a KNN classifier does not require a proper training

stage but only a collection of a certain number of reference examples. KNN is a discriminative classifier since it only provides decision boundaries between classes and does not require any modelling assumption. Nevertheless, to guarantee good performance, it is necessary to address a few issues related to the data and the distance used. Some of these issues are briefly reviewed below.

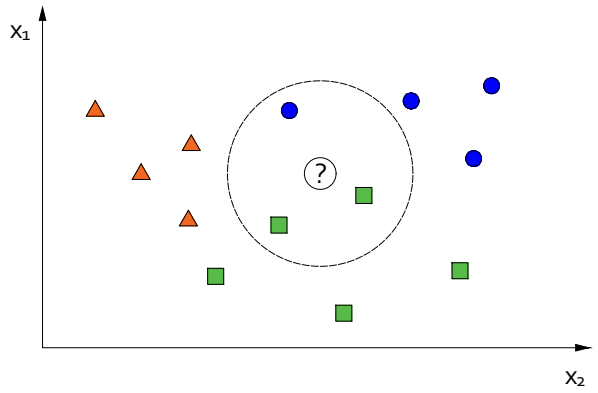


Figure 3.12: *Representation of K-nearest neighbours algorithm in $\mathbb{R}^2$ for $K = 3$. The distance is calculated with the Euclidean norm.*

Skewed datasets - One common issue related to the data emerges when the dataset is unbalanced or *skewed*. In this case, some classes are over-represented by a large number of reference examples, while others appear only a few times. Since the former are more abundant, in the case of near or overlapping classes, the probability of having them included in the K-neighbours is higher even if they don't belong to the same class. A possible solution to this issue consists in performing the majority vote after multiplying each neighbour by a weight that is inversely proportional to the distance. This solution generally makes the *impostors* less relevant. At the same time, the actual rule used for the majority vote can be made unbalanced by accounting for the prior probability of each class [206]. Another approach consists of sub-sampling or over-sampling opportunely the reference set to rebalance the number of instances per class [207]. For all the strategies above, a supervised approach can be used.

Curse of dimensionality - A second issue related to the data concerns the dimension D. When calculating the distance (3.80), for example using the common

Euclidean norm, a small value of the distance requires each correspondent couple of components to have similar values. This is more difficult to achieve when D increases from a few components to large arrays of features. When D increases, the ratio of the closest distance to the average distance between the test and the reference points approaches one [208, p. 170]. This issue is usually referred to as the *curse of dimensionality* [197, p. 18] and can be thought of as due to the fact that the same number of points N is sparser in a higher-dimensional space. Therefore, instead of increasing exponentially the amount of data, a possible solution comes from dimensionality reduction techniques such as PCA. Note that reducing the dimensionality helps also to eliminate those features that are irrelevant and potentially noisy.

Size of the dataset - KNN performs classification by comparing each test example with each reference example. Hence, to reduce the amount of memory and calculation required, it is useful to reduce the number of reference points without affecting the classification performance. This operation may also be beneficial in terms of noise reduction for the overall classification process [209]. Usually, a data reduction algorithm firstly eliminates the outliers[12] from the training set. Then, from the remaining examples, it tries to learn (*prototype selection*) or to generate (*prototype generation*) a set of *prototypes* that should be representative of the various clusters [210]. The prototypes should be able to correctly classify the remaining points, the *absorbed examples*, that can be safely removed from the dataset. Guaranteeing the quality of the prototypes, that is their ability to represent and classify the various clusters, is the key to reducing the number of reference examples without affecting the performance of the classifier [209].

Data normalisation - From the definition given above, since classification depends on the distance and some distances depend on the scale, it may be useful to introduce some form of normalisation in those cases where relevant components vary in very different ranges of magnitude [211].

Choice of the distance - A common choice for the distance (3.80) is the Euclidean

[12] For KNN, an outlier is a point in $\mathbb{R}^D$ surrounded by other points belonging to different classes. Outliers may be caused by noise in the dataset but also by an inadequate set of features or skewed data.

norm, but a range of metrics have been proposed in the literature to estimate similarities between examples and improve the selection of the right neighbours [212],[213]. For instance, the $LMNN$ generalises the Euclidean norm by defining the distance as:

$$d_{x,r_i} = (\boldsymbol{x} - \boldsymbol{r}_i)^T \boldsymbol{M} (\boldsymbol{x} - \boldsymbol{r}_i), \tag{3.81}$$

where $\boldsymbol{M}$ is a positive semi-definite matrix to be learned. The previous definition returns the Euclidean norm if $\boldsymbol{M}$ is the identity matrix.

3.4.2 Support vector machines

Support vector machines (SVM) [86],[202, p. 319] have been introduced for classification and regression tasks. For classification, the basic idea behind this technique is finding optimal hyper-dimensional surfaces that can separate examples belonging to different classes. As shown below, this method is naturally formulated for binary classification problems where the two classes are linearly separable. Nevertheless, with a few observations, the same approach can be extended to multi-class classification problems where classes are only non-linearly separable and include a certain number of outliers. An important characteristic of SVMs is that separation boundaries rely on just a small subset of training examples called *support vectors*; this peculiarity, at least after the training stage, mitigates issues like those related to the size of the dataset seen for KNN.

3.4.2.1 The linear classifier

For the simplest form of SVM, it is assumed that a set of N training examples $\{\boldsymbol{x}_i, c_i\} \in \chi \times C$ is linearly separable. Here χ represents a set of features $\in \mathbb{R}^D$, and C represents a set of two classes, c_+ and c_-. The aim is to find a hyper-plane in $\mathbb{R}^{D-1}$ that can separate the two different clusters, $\chi_+ = \{\boldsymbol{x}_i | c_i = c_+\}$ and $\chi_- = \{\boldsymbol{x}_i | c_i = c_-\}$, and maximise the separation margin. With reference to figure 3.13, named H the separation boundary, the classification rule should classify $\boldsymbol{x}_i \in \chi_+$ all the $\boldsymbol{x}_i$ above H and $\boldsymbol{x}_i \in \chi_-$ all the $\boldsymbol{x}_i$ below. Assuming $\boldsymbol{w}$ as a vector

orthogonal to H, the projection of a vector $\boldsymbol{x}$ on $\boldsymbol{w}$ is given by the inner product $\boldsymbol{w}^T\boldsymbol{x}$. Hence, the equation of H is given by:

$$\boldsymbol{w}^T\boldsymbol{x} + b = 0, \tag{3.82}$$

where both $\boldsymbol{w}$ and b need to be determined. Since data has been assumed linearly separable, it is possible to imagine H as equidistant from two margin boundaries, H_+ and H_-, which are two hyper-planes respectively placed on the closest examples of the two classes. The distance m_H between H_+ and H_- is the margin that SVM aims to maximise.

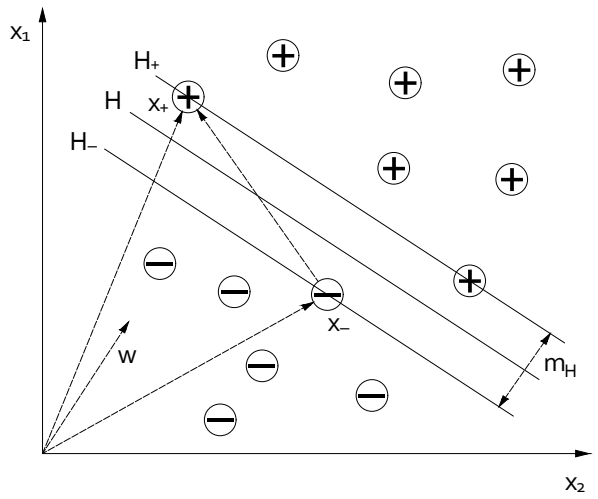


Figure 3.13: *Binary linear support vector machine. Construction of the optimised boundary margins in $\mathbb{R}^2$. H indicates the classification hyper-plane while H_+ and H_- are the optimised boundaries for the two linearly separable classes c_+ and c_-.*

Without loss of generality, since $\boldsymbol{w}$ and b need to be determined and margin boundaries are equidistant from H, it is possible to assume that the training examples satisfy the relations:

$$\boldsymbol{w}^T\boldsymbol{x}_i + b \geq 1 \qquad \forall\, \boldsymbol{x}_i \in \chi_+ \qquad \boldsymbol{w}^T\boldsymbol{x}_i + b = 1 \qquad \forall\, \boldsymbol{x}_i \in H_+ \tag{3.83a}$$

$$\boldsymbol{w}^T\boldsymbol{x}_i + b \leq -1 \qquad \forall\, \boldsymbol{x}_i \in \chi_- \qquad \boldsymbol{w}^T\boldsymbol{x}_i + b = -1 \qquad \forall\, \boldsymbol{x}_i \in H_-. \tag{3.83b}$$

Since C defines only two classes, for all $\boldsymbol{x}_i \in \chi$, it is possible to define $y_i = 1$ if $c_i = c_+$ and $y_i = -1$ if $c_i = c_-$. Using this definition together with relations (3.83) yields:

$$y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1 \quad \forall\, \boldsymbol{x}_i \in \chi \qquad y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) = 1 \quad \forall\, \boldsymbol{x}_i \in H_+, H_- \quad . \tag{3.84}$$

With reference to figure 3.13, the margin distance between H_+ and H_- can be written as:

$$m_H = \frac{\boldsymbol{w}^T}{\|\boldsymbol{w}\|_2}(\boldsymbol{x}_+ - \boldsymbol{x}_-) = \frac{2}{\|\boldsymbol{w}\|_2}, \tag{3.85}$$

where $\boldsymbol{x}_+$ and $\boldsymbol{x}_-$ are any two vectors on H_+ and H_- respectively, and $\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2}$ is a unitary vector orthogonal to the boundaries. The last identity is obtained by using (3.84) for $\boldsymbol{x}_+$ and $\boldsymbol{x}_-$. Hence, the SVM boundary maximisation problem is equivalent to the *primal optimisation problem*:

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|_2^2 \qquad with \qquad y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1 \quad \forall\, \boldsymbol{x}_i \in \chi, \tag{3.86}$$

the solution of which yields the *optimal margin classifier*[13]. This problem can be addressed numerically by using a *quadratic programming solver*, but it is usually further simplified by means of the *generalised Lagrangian multipliers* method[14]. This latter approach, in fact, offers some useful insights into the solution and a strategy to turn SVMs into non-linear classifiers.

The *generalised Lagrangian* for the problem (3.86) can be written as [214]:

$$\mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|_2^2 - \sum_{i=1}^{N} \alpha_i \Big(y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 \Big), \tag{3.87}$$

where α_i are the *Lagrangian multipliers* associated with each example $\boldsymbol{x}_i$. The

[13] In (3.86) the ^{2} is introduced only for mathematical convenience since it does not alter the original maximisation problem.

original primal optimisation problem (3.86) is equivalent to maximising (3.87) in respect to the multipliers α_i while satisfying the following *Karush-Kuhn-Tucker* (KKN) conditions [214]:

$$\frac{\partial \mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\alpha})}{\partial \boldsymbol{w}} = 0 \tag{3.88a}$$

$$\frac{\partial \mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\alpha})}{\partial b} = 0 \tag{3.88b}$$

$$\alpha_i g_i(\boldsymbol{w}) = 0 \qquad \forall\, i \in [1:N] \tag{3.88c}$$

$$g_i(\boldsymbol{w}) \leq 0 \qquad \forall\, i \in [1:N] \tag{3.88d}$$

$$\alpha_i \geq 0 \qquad \forall\, i \in [1:N], \tag{3.88e}$$

where $g_i(\boldsymbol{w}) = -y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) + 1$ and $\boldsymbol{\alpha} = [\alpha_1, ..., \alpha_N]$. The previous relationships are valid at the solution $\bar{\boldsymbol{w}}, \bar{b}, \bar{\boldsymbol{\alpha}}$.

This equivalent formulation is usually referred to as the *dual optimisation problem*. Equation (3.88c) implies that the only non-null α_i are those where $g_i(\boldsymbol{w}) = -y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) + 1 = 0$, that is only the points belonging to the margins H_+ and H_-. Then, (3.88a) and (3.88b) yield respectively:

$$\frac{\partial \mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\alpha})}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i = 0, \tag{3.89}$$

that is:

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i, \tag{3.90}$$

and:

$$\frac{\partial \mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\alpha})}{\partial b} = \sum_{i=1}^{N} \alpha_i y_i = 0. \tag{3.91}$$

An important consequence of (3.90) is that $\boldsymbol{w}$ can be simply found as a linear

[14] The generalised Lagrangian multipliers method extends the Lagrangian multipliers method when some inequalities are given for the constraints.

combination of $\boldsymbol{x_i}$ and, since α_i are not null only on the boundaries, $\boldsymbol{w}$ depends only on a small subset of examples placed on the optimised boundaries. The examples belonging to this subset are called the *support vectors*. Plugging (3.90) and (3.91) into (3.87) yields:

$$\begin{aligned}
&\mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\alpha}) \qquad (3.92)\\
&= \frac{1}{2}\sum_{j=1}^{N}\alpha_j y_j \boldsymbol{x}_j^T \sum_{i=1}^{N}\alpha_i y_i \boldsymbol{x}_i - \sum_{i=1}^{N}\alpha_i y_i \sum_{j=1}^{N}\alpha_j y_j \boldsymbol{x}_j^T \boldsymbol{x}_i - b\sum_{i=1}^{N}\alpha_i y_i + \sum_{i=1}^{N}\alpha_i \\
&= \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{j=1}^{N}\sum_{i=1}^{N}\alpha_j \alpha_i y_j y_i \boldsymbol{x}_j^T \boldsymbol{x}_i.
\end{aligned}$$

Hence, the simplified optimisation problem to be determined numerically consists in maximising:

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{j=1}^{N}\sum_{i=1}^{N}\alpha_j \alpha_i y_j y_i \boldsymbol{x}_j^T \boldsymbol{x}_i \quad with \quad \sum_{i=1}^{N}\alpha_i y_i = 0; \quad \alpha_i \geq 0 \quad \forall i \in [1:N]. \tag{3.93}$$

Once the multipliers α_i have been determined, the constant b can be found by using the (3.84) and evaluating $\boldsymbol{w}^T\boldsymbol{x}_i$ on H_+ or H_-, that is, using either the minimum value of $\boldsymbol{w}^T\boldsymbol{x}_i$ if $c_i = c_+$ or the maximum value of $\boldsymbol{w}^T\boldsymbol{x}_i$ if $c_i = c_-$. Finally, the *decision rule* for an unclassified example $(\boldsymbol{x}, c)$ can be written as:

$$if \qquad \boldsymbol{w}^T\boldsymbol{x} + b = \sum_{i=1}^{N}\alpha_i y_i \boldsymbol{x}_i^T \boldsymbol{x} + b \geq 0 \qquad then \qquad c = c_+ \tag{3.94a}$$

$$else \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad c = c_-, \tag{3.94b}$$

where the classification is performed by calculating the *scores* $\boldsymbol{w}^T\boldsymbol{x} + b$. Note that the decision rule is written as a simple linear combination of support vectors.

3.4.2.2 Non-linear extension and soft boundaries

The core of the linear algorithm found in the previous section is given by the boundary margin maximisation problem (3.93) and by the decision rule (3.94). In both cases, the examples $\boldsymbol{x}$ appear only in the form of the inner product $\boldsymbol{x}_j^T \boldsymbol{x}_i$. Recalling section 3.3.1.2, this property allows the use of the kernel trick also in the case of SVM. Again, given a Mercer kernel $\kappa(\boldsymbol{x}_j, \boldsymbol{x}_j)$ associated with a map function $\boldsymbol{\xi}(\boldsymbol{x})$, the classification in the higher dimensional space is obtained by replacing $\boldsymbol{x}_j^T \boldsymbol{x}_i$ with $\kappa(\boldsymbol{x}_j, \boldsymbol{x}_j)$ in (3.93) and (3.94). As for PCA, the main issue remains the selection of a useful kernel for a specific classification problem.

Although a non-linear implementation can improve the separability of different clusters, an important issue of the solution described in the previous section is that it does not allow for outliers in the data[15]. Indeed, an outlier does not satisfy the condition $y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \geq 1$. Furthermore, even when the clusters are separable, since SVMs strongly depend on just a few support vectors, a small number of examples placed near the boundary but far from the rest of the cluster can significantly affect the choice of the hyper-plane and the overall performance of the classifier. The above solution is then referred to as SVM with *hard boundaries*.

A commonly used workaround consists in introducing a *regularisation term* in (3.86) whose magnitude is controlled by a regularisation constant $C \geq 0$, that is:

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{N} \upsilon_i \qquad with \qquad \upsilon_i \geq 0 \quad y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \geq 1 - \upsilon_i \quad \forall i \in [1:N]. \tag{3.95}$$

In the relation above, an outlier is associated with a $\upsilon_i > 0$ and affects the minimisation by adding a positive term $C\upsilon_i$ to the objective function. The constant C modulates the regularisation. Since the formulation above allows outliers, it is usually referred to as SVM with *soft boundaries*. As for the hard boundaries, the solution to the (3.95) can be found by solving the associated dual problem. The generalised Lagrangian can be written as:

[15] For SVM, an outlier is an example that lies on the wrong side of the boundary.

$$\mathscr{L}(\boldsymbol{w}, b, \boldsymbol{\upsilon}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \tag{3.96}$$
$$= \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{N}\upsilon_i - \sum_{i=1}^{N}\alpha_i\Big(y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 + \upsilon_i\Big) - \sum_{i=1}^{N}\beta_i\upsilon_i,$$

and the KKN conditions require $\partial\mathscr{L}/\partial\boldsymbol{w}$ = 0, $\partial\mathscr{L}/\partial b = 0$, $\partial\mathscr{L}/\partial\upsilon_i = 0$ and:

$$\beta_i \geq 0 \quad -\upsilon_i \leq 0 \quad -\beta_i\upsilon_i = 0 \tag{3.97a}$$
$$\alpha_i \geq 0 \quad -y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) + 1 - \upsilon_i \leq 0 \quad -\alpha_i\Big(y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 + \upsilon_i\Big) = 0, \tag{3.97b}$$

$\forall\ i \in [1:N]$ and for $\boldsymbol{w}, b, \boldsymbol{\upsilon}, \boldsymbol{\alpha}, \boldsymbol{\beta}$ evaluated at the solution. By calculating the derivatives and applying conditions (3.97), the simplified Lagrangian to be maximised can be written as:

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{j=1}^{N}\sum_{i=1}^{N}\alpha_j\alpha_i y_j y_i \boldsymbol{x}_j^T\boldsymbol{x}_i \tag{3.98}$$
$$with \quad \sum_{i=1}^{N}\alpha_i y_i = 0; \quad 0 \leq \alpha_i \leq C \quad \forall\ i \in [1:N].$$

Note that (3.98) differs from (3.93) only for the upper boundary on α_i and that relation (3.90) remains valid.

Training a SVM with standard solvers generally requires a computational complexity proportional to N^3 [197, p. 499]. Several solutions have been proposed to make the training process more efficient [215], sometimes at the expense of accuracy [216]. On the other hand, some solvers can achieve better performance in the case of a large number of examples [217] or when the dimension of the features is high [218]. Further improvements can be achieved by optimising the data. For instance, to reduce the number of examples, the optimisation can be performed on a subset of randomly selected points [219], after analysing the clusters [220], or on those vectors that are more likely to be support vectors [221].

3.4.2.3 Multi-class SVM

The algorithm described in the two previous sections implements a classifier for a set of only two classes. However, extension to M classes is possible, and it is usually achieved with methods such as the *one versus all* and the *one versus one* [222].

In the *one versus all* approach, M different classifications are performed by using M different boundaries obtained by separating each of the M classes from the rest of the data. The simplest form of classification for the unknown example $\boldsymbol{x}$ is then obtained by evaluating the margin value $\boldsymbol{w}_m^T\boldsymbol{x}+b_m \; \forall \; m \in [1:M]$ and assigning the label of the class with the highest margin.

In the *one versus one* approach, $M(M-1)/2$ different classifications are performed by using $M(M-1)/2$ different boundaries obtained by grouping classes in pairs. In its simplest implementation, a vote is assigned to a class every time a binary classification is performed. The class of the unknown example $\boldsymbol{x}$ is then assumed to be the one with the majority of the votes.

Note that the sets of data used in the one versus all are generally unbalanced by construction and, unfortunately, the performance of SVMs decreases under these circumstances [223]. Possible solutions consist in rebalancing the data before training the machine [224] or modifying the training algorithm to account for the prior probability [225]. These solutions can also be adopted in case of an unbalanced dataset.

On the other hand, the majority vote described for the one versus one approach can introduce uncertainty regions where more than one class receives the majority of the votes. A better solution [226] (used also for the results reported in chapter 6) consists in defining a *binary loss* l_b as a function of the j^{th} binary classifier, of the k^{th} class, and of the value of the score $s_j(\boldsymbol{x}) = \boldsymbol{w}_j^T\boldsymbol{x}+b_j = \sum_{i=1}^{N}\alpha_{ji}y_{ji}\boldsymbol{x}_{ji}^T\boldsymbol{x}+b_j$:

$$l_b(\boldsymbol{x},j,k) = -g(\bar{y}_{jk}, s_j(\boldsymbol{x})), \tag{3.99}$$

where $\bar{y}_{jk}$ is $+1$ if the class k is the positive class of the binary classifier j, -1 if

it is the negative one, and zero if the binary classifier j is not trained for the class k. The classification label for $\boldsymbol{x}$ is then assigned by calculating the *negative loss*, that is averaging $l_b(\boldsymbol{x},:,k)$ as:

$$l(\boldsymbol{x},k) = \frac{\sum_j |\bar{y}_{jk}| l_b(\boldsymbol{x},j,k)}{\sum_j |\bar{y}_{jk}|}, \tag{3.100}$$

and choosing the class k with the maximum average. Frequently, the function $g(y,s)$ is assumed to be the *hinge* function:

$$g(y,s(\boldsymbol{x})) = max(0,1-ys(\boldsymbol{x})), \tag{3.101}$$

which is always positive except for those $\boldsymbol{x}$ on the correct side of the margin boundary $H+/H_-$, where it assumes a null value.

3.5 Assessing the performance

When performing automatic detection/classification tasks, it is essential to compare the results against different solutions/benchmarks. This operation is not always straightforward since different definitions of the evaluation indicators may provide very different results. Therefore, it is necessary to introduce some standard indexes and understand what kind of insight a specific index actually provides. Although some indicators are more robust than others, no definition suits all possible situations. The choice of a specific indicator should be motivated by the performance that is intended to be measured [38, p. 165].

3.5.1 Metrics

The definition of performance indicators is crucial to assess how good a certain machine is for the task it has been designed for. In general, different indicators offer different insights, and the numerical values provided may be misleading if the corresponding definition is not correctly interpreted. Besides, even when a certain indicator provides the desired information, it is important to consider the conditions

under which its numerical value has been determined. For instance, a *training set* too small might generate problems of *underfitting*, and an indication of bad performance might be more related to the amount of training data used rather than the machine itself. On the contrary, if the machine is designed to fit every oddity in the data or if train and test data are not well separated, problems of *overfitting* might generate unreliable good results with large discrepancies between train, test and field validation [227, p. 391]. Moreover, cumulative indicators can be defined as the average of the performances assessed per class. If the dataset is not well balanced, the performance of classes that are rare and potentially not very important might disproportionately affect the final rate.

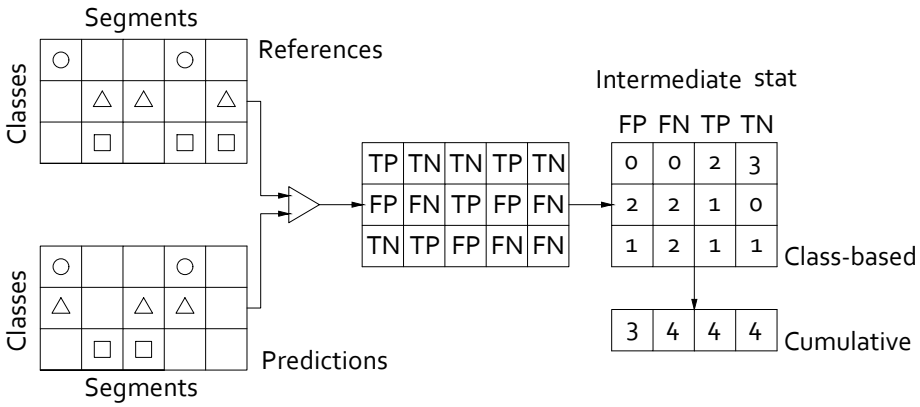


Figure 3.14: *Calculation of the intermediate statistics (segment-based). Metrics can be calculated from class-based or class-cumulative values.*

When performing audio event detection, it is possible to define *segment-based* or *event-based* metrics [228]. With the segment-based approach (figure 3.14), signals are divided into short segments, and elementary checks are performed per each class and each segment. With the event-based approach (figure 3.15), elementary checks are performed per each class and each matched/unmatched event. Working at the event level also means finding the start and the end time of each event to define the working boundaries. Certainly, it is always possible to merge several segments to provide metrics at the event level, but the minimum possible misalignment is dictated by the step resolution of the segments.

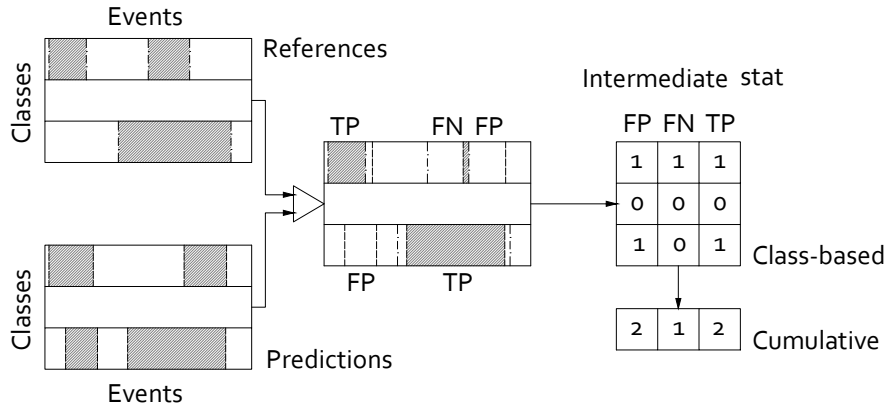


Figure 3.15: *Calculation of the intermediate statistics (event-based). Values of TP, FP, FN also depend on the alignment margin defined. TN are not defined since there is no margin to refer to.*

Metrics are usually defined as aggregates of results of elementary *trials* performed at the atomic level (segments or events). Each elementary trial assesses the prediction of the machine for a specific class c against the reference labels of an atom (Figures 3.14, 3.15). Four possible outcomes are possible [38, p. 167]:

- *TP - True positive*: both the machine and the reference labels indicate that c is contained in the atom;
- *TN - True negative*: both the machine and the reference labels indicate that c is not contained in the atom;
- *FP - False positive (I - insertion)*: the machine indicates that c is contained in the atom while the reference labels do not;
- *FN - False negative (D - deletion)*: the machine indicates that c is not contained in the atom while the reference labels do.

The collections of these atomic results are usually referred to as *intermediate statistics*. Attention should be paid to their interpretation. For instance, if the analysis is performed on segments, the number of TN is usually relatively high since a $+1$ is counted every time a class is not included and is not detected. On the other hand, if the analysis is performed on events, boundaries cannot be defined for a non-existing event, and TN cannot be defined. Moreover, in case of wrong classifications, a false positive and a false negative are generated at the

same time. This issue can be accounted for either as two separate errors or as a single error (a *S - substitution*).

Aggregates of intermediate statistics can be calculated by applying a certain metric operation over the whole set of results together (*instance-based*), or by calculating the average of the metrics evaluated per every single class (*class-based*). In the first case, the most common classes mainly determine the final value of the performance, while in the second case, the importance of rare classes might be disproportionately magnified. The definition of a few cumulative metrics is given below:

The *accuracy* is defined as:

$$Acc = \frac{TP + (TN)}{TP + (TN) + FP + FN} \tag{3.102}$$

and it measures the rate of the total correct predictions over the total number of predictions. TN are included only for segment-based analysis. Accuracy is not very meaningful in the case of sparse labels. Indeed, if a class is rare and the machine produces no output ($TP = FP = 0$), TN is much bigger than FN, and the accuracy for the class is roughly unitary even if the machine makes no predictions. Note that in certain situations, distinguishing between the error associated with FP and FN might be relevant, but this information cannot be extracted from the accuracy.

The *error rate* is defined as:

$$Err = \frac{D + I + S}{N}, \tag{3.103}$$

where N is the total count of reference labels or events. FP and FN are counted only once in case of a substitution S. Again, note that the error rate might be misleading. Indeed, a machine that makes no prediction returns a unitary error rate, while a machine that makes many insertions and, at the same time, a good number of correct predictions can have $Err > 1$.

The *F-score* [229, p. 119] is defined as:

$$Fsc = \frac{2}{1/P + 1/R} = \frac{2PR}{P+R}, \tag{3.104}$$

where the *precision P* is defined as $P = TP/(TP+FP)$ and the *recall R* is defined as $R = TP/(TP+FN)$. The maximum value of F-score is limited to 1 in the best case of no errors. Note that, in class-based analysis, each class should be included at least once in order to always return meaningful values.

For all the metrics above, in the case of event-based analysis, it is also necessary to define a maximum allowed misalignment for a detection to be considered valid.

3.6 Summary

This chapter analyses the prerequisites for the development of the dataset reported in chapters 5 and 6. It also introduces the necessary concepts for implementing the machine learning processing chain adopted in chapter 6, and provides a comprehensive literature review of the required notions. Three different signal representations, namely the short-time Fourier transform, the mel-frequency cepstral coefficients, and the wavelet scattering transform, are examined along with their representation in the time-frequency plane. The required properties of the acoustic features are described to understand how to tailor the extraction process and find the right balance among divergent necessities. In particular, feature invariance and stability are introduced, and the trade-off against loss of information and computational complexity is analysed. The potential of mel-frequency cepstral coefficients and wavelet scattering transform is discussed concerning the loss of information and computational complexity. Simple numerical simulations also illustrate the concept of stability. Moreover, the importance of feature normalisation is briefly discussed. The principal component analysis, a popular tool for feature transformation and reduction, is presented along with its non-linear kernel version. The potential benefits of feature dimensionality reduction and the limitations of non-linear kernels are highlighted. Regarding the classifiers, K-nearest neighbours

and support vector machines are described along with other issues, such as the curse of dimensionality, skewed dataset, dataset size, data normalisation, and distance definition. The classification margin for support vector machines is also introduced for further developments in chapter 6. Finally, three popular performance indicators, namely error rate, accuracy, and F-score, are reviewed, and the procedure for their calculation from intermediate statistics is given.

Chapter 4

An acoustic model for elastic pipes

This chapter introduces the acoustic model of the synthesiser. The model is articulated in three layers and allows the simulation of acoustic reverberations in elastic pipes with the setup illustrated in figure 4.1. The first layer accounts for the shape and the materials of the waveguide. It describes the dispersive behaviour of the modal propagation, providing information about phase and group velocity per each mode. The second layer describes the source S_P and how it is integrated into the model. It accounts for the geometry, the position and the dynamics of the source. Two different methods are analysed and compared. The third layer determines the output pressure at a point G once the input signal driving the source is known. Thanks to its layered structure, the model does not need to be entirely recalculated every time a new simulation is performed. For instance, if the waveguide and the source geometry remain unchanged, the output signal can be quickly recalculated, leaving the first two layers unaltered. As for any mathematical abstraction, the model accounts only for ideal components, and, in a real implementation, many aspects might differ from what is described here. Geometric flaws, materials characterisation, and mechanical implementation of source and receiver are just some examples [230],[231]. Besides, the range of geometrical options is limited and numerical finite element techniques are certainly more adequate for modelling complex shapes [232],[233]. Nevertheless, accounting for the limitations, the model provides a powerful tool when integrated into the

synthesizer described in chapter 5, giving an indication of the distortion caused by in-pipe propagation. Moreover, the model can be matched to simple, inexpensive test rigs for direct in-house experiments and measurements. The procedure to conduct some relevant measurements is provided later in this chapter.

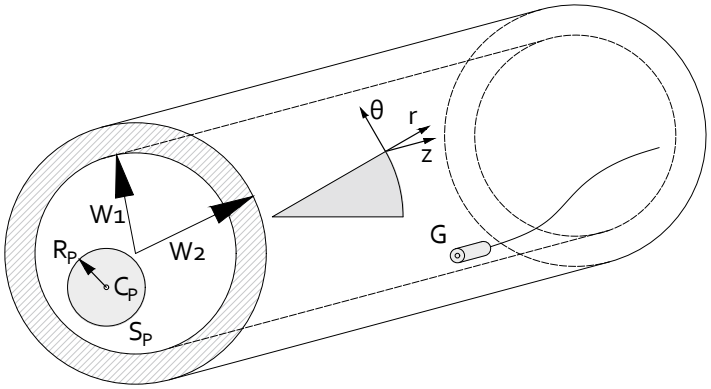


Figure 4.1: *Elastic waveguide with pressure source and receiver.*

4.1 Dispersive effects in waveguides with elastic walls

Section 2.8 describes how to model a signal propagating along a waveguide with rigid walls for a source whose distribution lies on a plane orthogonal to the main axis. It was shown that phase velocity and group velocity are affected by the boundary and how this issue translates into dispersive effects. In this section, the analysis of the dispersive effects is extended to the case of elastic boundaries, leaving for the following sections the problem of modelling sources and signal propagation. Given the importance in practical applications, we focus on circular pipes. It is remarked that the analytical approach is feasible only when the geometry allows a closed form of the equations (e.g. rectangular, circular, and elliptical sections). On the other hand, the meaningfulness of the results provided remains valid for pipes of arbitrary shapes. The reference setup is an aluminium pipe (inner radius $W_1 = 49.20mm$, outer radius $W_2 = 50.80mm$) filled with inviscid water. Further examples describing how the dispersion changes with

geometry and materials are mentioned in section 4.1.2. The values of all the physical quantities used are reported below:

Material	Density $[kg/m^3]$	Longitudinal Velocity $[m/s]$	Shear Velocity $[m/s]$
Water	1000	1479	-
Aluminium	2700	6420	3040
PMMA	1190	2690	1340

Table 4.1: *List of material properties used for the acoustic models.*

4.1.1 Determination of propagative and evanescent modes

In section 2.9, it has been shown that the homogeneous Helmholtz equations for the potentials (2.155) with boundary conditions (2.176) admit non-trivial solutions for those values of k_z for which the determinant of the matrix $\boldsymbol{D}$ given by (2.179) is null. It is remarked that equations (2.155) refer to the potentials. In the inner liquid domain, where the pressure values are meant to be measured, the potential ϕ is related to the pressure p by the (2.46), which in the harmonic regime can be written as:

$$\hat{p} = \rho \omega^2 \hat{\phi}, \tag{4.1}$$

meaning that equivalent results can be obtained for the potential or the pressure.

4.1.1.1 Numerical calculation of the axial wavenumbers

Due to the analytical complexity of the determinant $|\boldsymbol{D}|$, k_z needs to be determined numerically and in a predefined range of frequencies. This is an important difference compared to the case of rigid boundaries where an analytical solution can be worked out. Given the mode order n and the frequency ω, the algorithm implemented (algorithm 1) aims to find the M values of k_z that satisfy the equation $|\boldsymbol{D}(n,\omega,k_z)| = 0$ in a predefined range for k_z. These values identify the modes (n,m), $\forall\, m \in [1:M]$, at frequency ω.

Algorithm 1 Simplified numerical k_z solver algorithm for propagative modes

Input: $\boldsymbol{D}(n,\omega,k_z)$, N, $\boldsymbol{V}_\omega$, $\boldsymbol{V}_{k_z}$, S_ω^{init}, $\boldsymbol{\Omega}^{init}$, R_{k_z}, W_ω, W_{k_z}

Output: $\boldsymbol{\omega}_{cut}$, $\boldsymbol{K}$

$\boldsymbol{V}_\omega^{cut} \leftarrow \boldsymbol{V}_\omega \uparrow 10$ ▷ Increase $\boldsymbol{V}_\omega$ resolution by 10

$\boldsymbol{V}_\omega^{init} \leftarrow \boldsymbol{V}_\omega \downarrow S_\omega^{init}, \boldsymbol{\Omega}_{init}$ ▷ Obtain initialisation frequencies

for $n = 0 : N$ **do** ▷ Repeat for all the modes

  for $c_\omega = 1 : length(\boldsymbol{V}_\omega^{cut})$ **do** ▷ Scan for $k_z = 0$ cut frequencies

    $\boldsymbol{A}^{cut} \leftarrow |\boldsymbol{D}(n, \boldsymbol{V}_\omega^{cut}{}_{(c_\omega)}, 0)|$ ▷ Save $|\boldsymbol{D}|$ for $\boldsymbol{V}_\omega^{cut}$ with $k_z = 0$

  end for

  $\boldsymbol{A}_\pm^{cut} \leftarrow sign(\boldsymbol{A}^{cut})$ ▷ Find where $\Re(\boldsymbol{A}^{cut})$ changes sign

  $\boldsymbol{\omega}_{cut} \leftarrow disc(\boldsymbol{A}_\pm^{cut}, W_\omega)$ ▷ Clean discontinuities, save cut frequencies

  $\boldsymbol{V}_\omega^{init} \leftarrow \boldsymbol{V}_\omega^{init}, neighbours(\boldsymbol{\omega}_{cut})$ ▷ Add points around cut frequencies

  $\boldsymbol{L}_l, \boldsymbol{L}_r \leftarrow steps(\boldsymbol{V}_\omega, \boldsymbol{V}_\omega^{init})$ ▷ Define scan intervals $\forall\, \boldsymbol{V}_\omega^{init}$ steps

  for $i_\omega = 1 : length(\boldsymbol{V}_\omega^{init})$ **do** ▷ Init scan starting from $\boldsymbol{V}_\omega^{init}$

    for each x *in* $\boldsymbol{V}_{k_z}$ **do** ▷ Scan $\forall x \in \boldsymbol{V}_{k_z}$

      $\boldsymbol{A}^{init} \leftarrow |\boldsymbol{D}(n, \boldsymbol{V}_\omega^{init}{}_{(i_\omega)}, x)|$ ▷ Save $|\boldsymbol{D}|$ for $\forall x \in \boldsymbol{V}_{k_z}$

    end for

    $\boldsymbol{A}_\pm^{init} \leftarrow sign(\boldsymbol{A}^{init})$ ▷ Find where $\Re(\boldsymbol{A}^{init})$ changes sign

    $s_{\omega_i} \leftarrow s_{\omega_i} \big| \boldsymbol{V}_\omega^{init}{}_{(i_\omega)} == \boldsymbol{V}_{\omega(s_{\omega_i})}$ ▷ Select scan init index in $\boldsymbol{V}_\omega$

    $\boldsymbol{K}_{l/r(:,s_{\omega_i})} \leftarrow disc(\boldsymbol{A}_\pm^{init}, W_{k_z})$ ▷ Clean $\boldsymbol{A}_\pm^{init}$, save k_z in $\boldsymbol{K}_l$, $\boldsymbol{K}_r$

    for $s_\omega = s_{\omega_i} - 1 : s_{\omega_i} - \boldsymbol{L}_{l(i_\omega)}$, $\quad s_\omega = s_{\omega_i} + 1 : s_{\omega_i} + \boldsymbol{L}_{r(i_\omega)}$ **do** ▷ Scan

      for each k_z *in* $\boldsymbol{K}_{l(:,s_\omega+1)}$, $\; k_z$ *in* $\boldsymbol{K}_{r(:,s_\omega-1)}$ **do** ▷ Known k_z

        for $x = k_z - R_{k_z} : k_z + R_{k_z}$ **do** ▷ Scan around known k_z

          $\boldsymbol{A} \leftarrow |\boldsymbol{D}(n, \boldsymbol{V}_{\omega(s_\omega)}, x)|$ ▷ Save $|\boldsymbol{D}|$ for the neighbours

        end for

      end for

      $\boldsymbol{A}_\pm \leftarrow sign(\boldsymbol{A})$ ▷ Find where $\Re(\boldsymbol{A})$ changes sign

      $\boldsymbol{K}_{l/r(:,s_\omega)} \leftarrow disc(\boldsymbol{A}_\pm, W_{k_z})$ ▷ Clean $\boldsymbol{A}_\pm$, save k_z in $\boldsymbol{K}_l$, $\boldsymbol{K_r}$

    end for

  end for

  $\boldsymbol{K} \leftarrow merge(\boldsymbol{K}_l, \boldsymbol{K}_r)$ ▷ Merge $\boldsymbol{K}_l$ and $\boldsymbol{K}_r$ into $\boldsymbol{K}$

  return $\boldsymbol{\omega}_{cut}$, $\boldsymbol{K}$

end for

Apart from $\boldsymbol{D}(n,\omega,k_z)$ in its analytical form, the main input variables fed to the algorithm are: the maximum mode order N, the frequency vector $\boldsymbol{V}_\omega$, and the vector of test points $\boldsymbol{V}_{k_z}$. Both $\boldsymbol{V}_\omega$ and $\boldsymbol{V}_{k_z}$ are chosen in the desired range and with the necessary resolution.

In the first step, the $k_z = 0$ *cut frequencies* are calculated by solving the equation $|\boldsymbol{D}(n,\boldsymbol{V}_\omega^{cut}{}_{(c_\omega)},0)| = 0$, with c_ω index of $\boldsymbol{V}_\omega^{cut}$. The solution returns the frequency values at which the branches of the modes intersect the frequency axis. Since these points mark the frequencies where evanescent modes turn propagative, higher accuracy is convenient. Therefore, $\boldsymbol{V}_\omega^{cut}$ is obtained from $\boldsymbol{V}_\omega$ increasing the resolution by a factor 10. The complex values of the determinant $|\boldsymbol{D}(n,\boldsymbol{V}_\omega^{cut}{}_{(c_\omega)},0)|$ are analysed to find the frequencies where the sing changes. For this purpose, the imaginary part does not add any relevant information and can be neglected. A further check is performed to remove those points where the change of sign is given by a $\pm\infty$ discontinuity. This refinement is achieved by inspecting the local maximum of the absolute value of $\Re\big(|\boldsymbol{D}(n,\boldsymbol{V}_\omega^{cut}{}_{(c_\omega)},0)|\big)$ in a neighbourhood of radius W_ω around the frequency where the sign changes. The remaining zeros are the $k_z = 0$ cut frequencies and are saved in $\boldsymbol{\omega}_{cut}$.

In the next step, the branches of the modes are calculated. The aim is to minimise the calculations required by shrinking the region where $|\boldsymbol{D}(n,\omega,k_z)|$ needs to be evaluated. The algorithm is initialised by defining a small subset of S_ω^{init} frequencies obtained from $\boldsymbol{V}_\omega$ at regular intervals. This subset, named $\boldsymbol{V}_\omega^{init}$, is further expanded by adding a few additional initialisation points in a neighbourhood of the frequencies saved in $\boldsymbol{\omega}_{cut}$. If desired, a set of user-defined steps $\boldsymbol{\Omega}^{init}$ can also be included. Then, the left and right scan intervals $\boldsymbol{L}_l$ and $\boldsymbol{L}_r$ are obtained accordingly. The initialisation frequencies mark the beginning of the scans, meaning that the first set of k_z is extracted by calculating $|\boldsymbol{D}(n,\boldsymbol{V}_\omega^{init}{}_{(i_\omega)},x)|$, with i_ω index of $\boldsymbol{V}_\omega^{init}$ and $\forall\, x \in \boldsymbol{V}_{k_z}$. This operation is represented in figure 4.2, where the initial $x = k_z$ solutions are marked by the red circles placed on the green lines at frequencies $\boldsymbol{V}_\omega^{init}{}_{(i_\omega)}$. From $\boldsymbol{V}_\omega^{init}{}_{(i_\omega)}$, the scan is continued to the left and to the right, setting the index s_ω according to the intervals defined in $\boldsymbol{L}_l$ and $\boldsymbol{L}_r$, respectively. The calculation of $|\boldsymbol{D}(n,\boldsymbol{V}_{\omega(s_\omega)},x)|$, however, is now executed only for

a small subset of $x \in \boldsymbol{V}_{k_z}$, focusing only on neighbourhoods of radius R_{k_z} around known k_z solutions found in the previous iteration steps. As for the cut frequencies, solutions of $|\boldsymbol{D}(n, \boldsymbol{V}_{\omega(s_\omega)}, x)| = 0$ are found by analysing the change of sign and checking for $\pm\infty$ discontinuities in a neighbourhood of radius W_{k_z}. In the last step, the matrices yielded by the left and right scans are merged in a unique matrix $\boldsymbol{K}$. Figure 4.2 illustrates the results of the solver for the $n = 2$ evanescent modes. Note how the extra initialisation points in the neighbourhood of the $k_z = 0$ cut frequency at $18.5kHz$ allow the discovery of the small branch not intercepted by the green line placed at $V_{\omega}^{init}{}_{(i_\omega)} = 20kHz$ (see also mode $(2, C)$ figure 4.9-B).

The resolution of the numerical analysis is a trade-off between computational cost and accuracy. Here, a $\Delta\omega$ of $5Hz$ for $\boldsymbol{V}_\omega$ is chosen to provide good frequency resolution and detect those branches with large $\Delta k_z/\Delta\omega$ variations (for instance see mode $(3,3)$ in figure 4.12-A). On the other hand, a Δk_z of $0.01 rad/m$ for $\boldsymbol{V}_{k_z}$ guarantees an accurate evaluation of the derivatives for the group velocities. Additional issues related to the finite resolution of the vectors $\boldsymbol{V}_\omega$ and $\boldsymbol{V}_{k_z}$ are discussed in the next section. Finally, the algorithm above (algorithm 1) is used to extract propagative modes. For evanescent modes, however, the procedure can be identically repeated by simply using $i\boldsymbol{V}_{k_z}$ in place of $\boldsymbol{V}_{k_z}$.

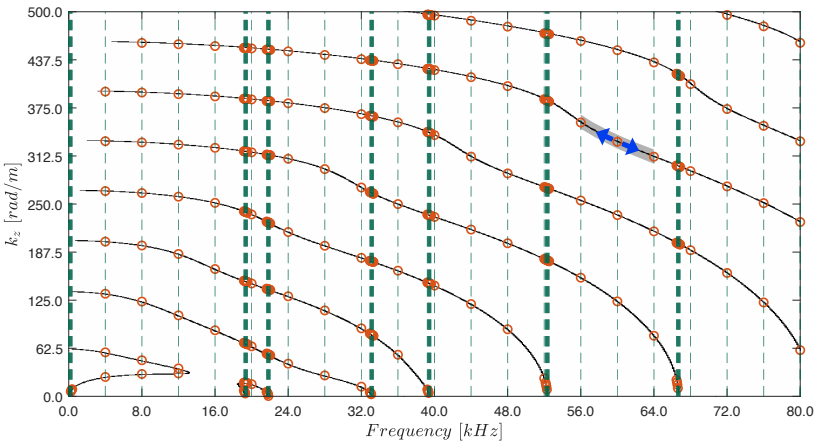


Figure 4.2: *Results of the k_z numerical solver for the $n = 2$ evanescent modes. The green dashed lines indicate the initialisation frequencies. The red circles mark the initial set of k_z from which the scan for the remaining k_z begins. The directions of the scan and the radius of the search neighbourhood for one of the initial points are indicated by the blue arrows and the grey-shaded region, respectively. Four initial neighbouring frequencies $50Hz$ apart are added at both sides of each $k_z = 0$ cut frequency.*

4.1.1.2 Mode separation

The execution of the solver described in the previous section returns a matrix $\boldsymbol{K}$ where the contained axial wavenumbers $k_{z_{nm}}(\omega)$ mark, for a given n and at a given frequency ω, up to M different modes. These modes, however, are not separated, and the extraction of additional information, such as phase and group velocities, requires further processing. In section 2.8, it has been shown that orthogonality provides major simplifications for a simple liquid cylinder with rigid boundaries. This benefit can also be exploited for other physical configurations. In [128], for instance, the orthogonality is at the foundation of the technique used to separate the modes in the case of an elastic multilayered cylinder. When modes are orthogonal, the inner product:

$$g\left(n',n'',k_{z_{n'm'}},k_{z_{n''m''}}\right) = \langle e^{in'\theta} J_{n'}(q_{n'm'}r),\, e^{in''\theta} J_{n''}(q_{n''m''}r)\rangle, \tag{4.2}$$

with $q_{nm}^2 = \omega^2/c^2 - k_{z_{nm}}^2$, is null for any $n' \neq n''$ or $m' \neq m''$ (see sections 2.8 and 2.9). For what concerns the separation of the modes discussed in this section, the index n can be neglected imposing $n = n' = n''$. Assuming that the resolution $\Delta\omega$ is sufficiently small and the orthogonality is verified as in [128], the h^{th} point of the solution at frequency $\omega + \Delta\omega$, $k_{z_{nh}}(\omega + \Delta\omega)$, is the continuation of the m^{th} mode at frequency ω, $k_{z_{nm}}(\omega)$, when the related inner product given by equation 4.2, $g\left(n,n,k_{z_{nm}}(\omega),k_{z_{nh}}(\omega+\Delta\omega)\right)$, is reasonably not null. This consideration offers a simple criterion for mode sorting. Nevertheless, named $\vec{\boldsymbol{v}}_{nm}$ and $\boldsymbol{\sigma}_{nm}$ the particle velocity and the stress tensor for mode (n,m), the demonstration proposed in [128], extension of a similar work in [129], relies on the reciprocity equation:

$$\nabla \cdot (\vec{\boldsymbol{v}}_{n'm'} \cdot \boldsymbol{\sigma}_{n''m''} - \vec{\boldsymbol{v}}_{n'm''} \cdot \boldsymbol{\sigma}_{n'm'}), \tag{4.3}$$

and requires, among other hypotheses, free external boundaries and continuity of the displacement. Although the same assumption for the external boundaries has been adopted in this work, the liquid-solid interface considered imposes only the continuity of the normal displacement, which is a weaker condition not

sufficient to preserve the orthogonality. Consequently, the inner product 4.2 cannot be assumed null for $m' \neq m''$, and separating points belonging to different modes requires a different approach.

Algorithm 2 Simplified mode separation algorithm for a set of M modes of order n

Input: $\boldsymbol{K}, \boldsymbol{V}_\omega, H, R_{k_z}$

Output: $\boldsymbol{K}_M$,

$\boldsymbol{K}_M \leftarrow fit(\boldsymbol{K}, \boldsymbol{V}_\omega)$ ▷ Organise k_z values by frequency step
for $h_\omega = 2 : length(\boldsymbol{V}_\omega)$ **do**
  $\boldsymbol{X} \leftarrow fill(\boldsymbol{K}_{M(:,h_\omega - H:h_\omega - 1)})$ ▷ Last part of the sorted branches
  $\boldsymbol{Y} \leftarrow \boldsymbol{K}_{M(:,h_\omega)}$ ▷ Get the next set of points to sort
  $M' \leftarrow branches(\boldsymbol{K}_{M(:,1:h_\omega - 1)})$ ▷ Total branches at step $h_\omega - 1$
  $\boldsymbol{X}_\delta \leftarrow diff(\boldsymbol{X})$ ▷ Get the differentials for the M' branches
  $\boldsymbol{W} \leftarrow [0 : 1/H : 1]$ ▷ Differential weights
  $\tilde{\boldsymbol{\Delta}}_y \leftarrow \boldsymbol{X}_\delta \boldsymbol{W}^T / sum(\boldsymbol{W})$ ▷ Averaged differentials for the M' branches
  $\tilde{\boldsymbol{y}} \leftarrow \boldsymbol{K}_{M(:,h_\omega - 1)} + \tilde{\boldsymbol{\Delta}}_y$ ▷ Predicted values for the next k_z of the branches
  for $m = 1 : M'$ **do**
    $j, l \leftarrow min(\tilde{\boldsymbol{y}}, \boldsymbol{Y})$ ▷ Closest points between $\tilde{\boldsymbol{y}}$ and $\boldsymbol{Y}$
    $e \leftarrow abs(\tilde{\boldsymbol{y}}_{(j)} - \boldsymbol{Y}_{(l)})$ ▷ Prediction error for the closest points
    if $e < R_{k_z}$ **then** ▷ The points are closer than the max radius
      $\boldsymbol{K}_{M(j,h_\omega)} \leftarrow \boldsymbol{Y}_{(l)}$ ▷ Set the point returning the min distance
      $\boldsymbol{Y}_{(l)} \leftarrow -\infty$ ▷ Remove the assigned value from the list
    end if
    $\tilde{\boldsymbol{y}}_{(j)} \leftarrow +\infty$ ▷ Remove the predicted value from the list
  end for
  $M'' \leftarrow sum(\boldsymbol{Y} > 0)$ ▷ Number of new branches
  $\boldsymbol{K}_{M(M'+1:M'+M'',h_\omega)} \leftarrow \boldsymbol{Y}(\boldsymbol{Y} > 0)$ ▷ Add new branches
end for
$\boldsymbol{K}_M \leftarrow clean(\boldsymbol{K}_M)$ ▷ Remove undesired branches
$\boldsymbol{K}_M \leftarrow interpol(\boldsymbol{K}_M)$ ▷ Close gaps by interpolation
return $\boldsymbol{K}_M$

The pseudo-code summarised in algorithm 2 describes the topological approach used as a workaround. The basic idea is to isolate the branches of the solution by joining together neighbouring points that ensure the continuity of the first derivative. Firstly, the matrix $\boldsymbol{K}$ obtained from the solver is reorganised into a new matrix $\boldsymbol{K}_M$ where the k_z values are arranged in columns by frequency step. In the main part of the algorithm, this new matrix is sorted to obtain a mode per each of its rows. Assuming $\boldsymbol{K}_M$ sorted up to the frequency $\boldsymbol{V}_{\omega(h_\omega-1)}$, the selection of the next point of a branch at step h_ω relies on the calculation of the predicted value obtained from the differential of the previous H points. In the pseudo-code, M' are the number of branches found at step $h_\omega - 1$, $\boldsymbol{Y}$ represents the points to sort organised in a column vector of size $M' \times 1$ extracted from $\boldsymbol{K}_M$ at step h_ω, and $\boldsymbol{X}$ is a matrix $M' \times H$ obtained from $\boldsymbol{K}_M$ by selecting the columns from step $h_\omega - H$ to $h_\omega - 1$. The rows of $\boldsymbol{X}$ with at least one value not null are completed by an extended linear interpolation. This solution also allows a trivial initialisation obtained by simply skipping the first frequency step. The M' approximated differentials $\tilde{\boldsymbol{\Delta}}_y$ are calculated as the weighted average of the differentials $\boldsymbol{X}_\delta$ of the M' rows of $\boldsymbol{X}$. The weights are linearly assigned, from 0 for the first item to 1 for the last. This operation provides a more accurate evaluation of the desired differential, reducing the importance of points farther apart. The predicted value $\tilde{\boldsymbol{y}}$ at step h_ω are then calculated as $\boldsymbol{K}_{M(:,h_\omega-1)} + \tilde{\boldsymbol{\Delta}}_y$. Finally, this prediction is compared against the values in $\boldsymbol{Y}$ to select the points $\tilde{\boldsymbol{y}}_{(j)}$ $\boldsymbol{Y}_{(l)}$ whose distance is minimum and below the maximum radius R_{k_z}. The last step is repeated for all the M' points in $\tilde{\boldsymbol{y}}$. The M'' points in $\boldsymbol{Y}$ not selected as branch continuation are added at the bottom of $\boldsymbol{K}_M$ as starting points of new branches.

When separating the branches, the procedure implemented deals with potential missing points in the solutions obtained from the solver. This issue is caused by the shape of the function $|\boldsymbol{D}(n,\omega,x)|$ and by the finite resolution of the vectors $\boldsymbol{V}_\omega$ and $\boldsymbol{V}_{k_z}$. From this point of view, two different cases have been detected while attempting to solve the equation $|\boldsymbol{D}(n,\omega,x)| = 0$. The first case is given by the change of sign associated with $\pm\infty$ discontinuities. These points are rejected by analysing the absolute value of $\Re(|\boldsymbol{D}(n,\omega,x)|)$ in the related neighbourhoods.

However, when a zero-cross point is too close to a discontinuity, the solution is neglected along with the latter. In the second case, generally verified for evanescent modes at low frequencies, the function $|\boldsymbol{D}(n,\omega,x)|$ is tangent to the null value, and the solver cannot detect the solution for k_z. In the matrix $\boldsymbol{K}_M$, the issues above cause small gaps in the branches of the modes. These gaps, however, are small enough to be reconstructed with a simple linear interpolation, and this task is covered by the last part of the algorithm.

Finally, the mode extractor also produces a table reporting the *cut-on* and the *cut-off* frequencies. The cut-on frequency of a mode, either propagative or evanescent, indicates the frequency at which a mode begins its contribution to the pressure field. Similarly, the cut-off is the frequency at which this contribution disappears. Some of the cut-on/off frequencies are just the $k_z = 0$ cut frequencies (vector $\boldsymbol{\omega}_{cut}$) at which modes convert from evanescent to propagative and vice versa (for instance, see figures 4.3). For other modes, however, the shape is more intricate, and a single branch can identify two different modes coexisting at the same frequency. In the latter case, the cut-on/off does not mark any evanescent/propagative transition (for instance, see figure 4.9-B, modes $(2,A)$ and $(2,B)$).

4.1.1.3 Discussion and analysis of a case study

Modes are identified by two indexes: n and m. The index n relates to the θ angular direction, while the index m relates to the r radial direction. Therefore, when the problem is axially symmetric, modes for $n \neq 0$ can be neglected. On the contrary, for a non-axisymmetric configuration, modes in the range $[-N:N]$ should be considered. Note that, N mainly depends on the potential field that needs to be described, and that, for cylindrical geometries, the sign of n is irrelevant. As a rule of thumb, the minimum value for N should be chosen so that the angular dimension of the smallest field feature is around π/N. If the potential field is smooth and changes slowly in the θ direction, a satisfactory truncation can be achieved even for $N = 3$. The index m accounts for the radial variations. For clarity, propagative and evanescent modes are marked using numbers and letters, respectively. Similarly to n, the optimal truncation for m depends on the variation of the potential field along r. For m, however, a few more observations are required.

n \ m	1	2	3	4	5	6	7	8	9
0	0 -	0.015 -	12.421 -	25.208 -	38.832 -	52.895 -	67.233 -	- -	- -
1	0 -	0 -	9.678 -	13.871 -	20.371 -	31.917 -	45.664 -	59.883 -	74.339 -
2	0 -	0.181 -	19.355 -	21.842 -	33.135 -	39.418 -	52.344 -	66.668 -	- -
3	0 -	0.572 -	27.680 27.743	27.680 -	29.032 -	42.415 -	51.779 -	59.261 -	73.324 -

Table 4.2: *Cut-on (top) and cut-off (bottom) frequencies in kHz for a circular aluminium tube filled with inviscid water. Propagative modes. $W_1 = 49.20mm$, $W_2 = 50.80mm$ (see figure 4.1). The indices n and m indicate the modal order for the angular and radial variations, respectively. Each box reports the cut-on frequency at the top and the cut-off frequency at the bottom. For those modes whose frequencies are beyond the range of the simulations, the "-" indicates the unknown value or existence.*

n \ m	A	B	C	D	E	F	G	H	I
0	0 12.421	0 25.208	0 38.832	0 52.895	0 67.233	0 -	0 -	- -	- -
1	0 9.678	0 13.871	0 20.371	0 31.917	0 45.664	0 59.883	0 74.339	0 -	0 -
2	0.181 13.210	0 13.210	18.555 19.355	18.555 21.842	0 33.135	0 39.418	0 52.344	0 66.668	0 -
3	0.572 16.420	0 16.420	27.743 29.032	0 42.415	0 51.779	0 59.261	0 73.324	0 -	0 -

Table 4.3: *Cut-on (top) and cut-off (bottom) frequencies in kHz for a circular aluminium tube filled with inviscid water. Evanescent modes. $W_1 = 49.20mm$, $W_2 = 50.80mm$ (see figure 4.1). As for table 4.2, the values at the top indicate the cut-on, the values at the bottom indicate the cut-off, and values beyond the range of the simulation are indicated with "-". For clarity, a literal sequence is used for the index m of the evanescent modes.*

Apart from the geometrical characteristics of the field, the required m modes also depend on the signal processed for a specific simulation. Indeed, as seen for the rigid wall case, modes propagate only above their cut-on frequency. Therefore, knowing the frequency range of interest, it is possible to omit all those modes whose cut-on frequency lies beyond the upper-frequency boundary without affecting the accuracy of the simulated propagation. In the non-propagative regions of the spectrum, however, the related evanescent modes should be considered to improve the accuracy of the modelled source. This approach, which is new to the best of our knowledge, is dictated by the lack of modal orthogonality and is further motivated in the following sections.

The results reported in the rest of this section mainly refer to the aluminium pipe of figure 4.1 with $W_1 = 49.20mm$ and $W_2 = 50.80mm$. Table 4.2 and 4.3 list cut-on and cut-off frequencies for both propagative and evanescent modes, respectively. The m index for the evanescent case is marked with letters since the relation between evanescent and propagative modes is not always one-to-one. Consider, for instance, the modes $(0,A)$ and $(0,3)$: the cut-off frequency of the evanescent mode $(0,A)$ matches the cut-on frequency of the propagative mode $(0,3)$. Both modes can be thought of as belonging to the same branch of the solution, which describes an exponentially decaying mode for $f < 12.421kHz$ and a propagative mode beyond the same frequency. This description is better understood from figure 4.3 where modes $(0,A)$ and $(0,3)$ are marked with the same colour and intersect the frequency axis at the same point. Consider now modes $(3,3)$, $(3,5)$ and $(3,C)$. As shown in figure 4.12, the evanescent mode $(3,C)$ intersects the frequency axis at two different points, which are also the intersection points of the propagative modes $(3,3)$ and $(3,5)$. Therefore, they all belong to the same branch of the solution. Interestingly, the same branch also includes mode $(3,4)$, which cannot be simply considered as the continuation of mode $(3,3)$. Indeed, these two propagation statuses can coexist in the frequency range between $27.675kHz$ and $27.742kHz$. In other words, although these modes can be visualised as belonging to the same branch, they are associated with different waves and different wavenumbers. A similar behaviour can also be

observed for evanescent modes $(2,A)$, $(2,B)$, $(2,C)$, $(2,D)$, $(3,A)$, $(3,B)$.

Figures 4.3, 4.6, 4.9, 4.12 show that a certain degree of similarity exists between dispersive effects for different values of n. Further observations can be made by recalling the definition given in section 2.7 for *phase velocity* and *group velocity*:

$$c_p = \frac{\omega}{k_z} \qquad c_g = \frac{\partial \omega}{\partial k_z}, \tag{4.4}$$

where ω is the angular frequency, and k_z is the axial wavenumber of the mode inspected. Interestingly, apart from $n = 0$, the modes $m = 1$ exhibit non-dispersive behaviour, always maintaining the proportionality between frequency and wavenumber. As it can be observed in figures 4.7, 4.8, 4.10, 4.11, 4.13, 4.14, for these modes, non-dispersive behaviour means constant phase and group velocity. The numerical value also matches the shear velocity of the external aluminium tube, meaning that the shear propagation is stimulated by non-axisymmetric excitation and is not affected by dispersion effects. For all the other modes, the dispersion introduces a non-linear behaviour for phase and group velocity. Excluding the propagative modes existing at very low frequencies, around the cut-on, the phase velocities decrease from an asymptotic infinite value, while the group velocities increase from zero. Therefore, around the cut-on, phase and energy propagate above and below the intrinsic material velocities, respectively. In other ranges, phase velocities show a flat interval around the shear velocity and an asymptotic convergence at the velocity of the inner fluid. Group velocities show a first oscillation (always below the longitudinal velocity of the aluminium shell), a flat interval around the shear velocity, and an asymptotic value equivalent to the one seen for the phase velocities. As for the rigid boundary, the asymptotic convergence indicates a straight propagation in the z direction at frequencies much higher than the cut-on [65, p. 255]. Only one mode per each group, namely $(0,1)$, $(1,2)$, $(2,2)$, $(3,2)$, exhibit different behaviour, and their group and phase velocities always remain below the longitudinal velocity of the inner liquid. Finally, note that under specific hypotheses, the characteristic equation (2.179) can assume particular forms where the motion exhibits special features. For instance, the motion obtained imposing $k_z = 0$ is analysed in [143],[234].

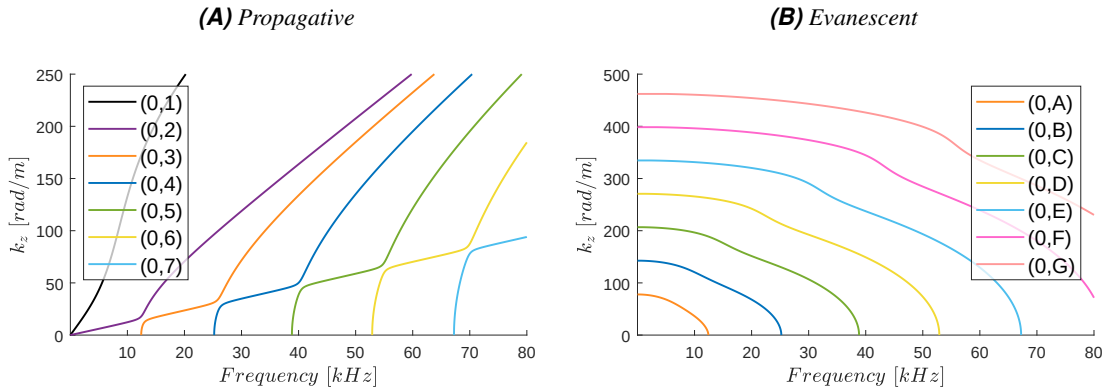


Figure 4.3: *Axial wavenumber $k_{z_{0m}}$ for an aluminium pipe filled with inviscid water. $n = 0$, $W_1 = 49.20mm$, $W_2 = 50.80mm$. The propagative branches (A) match the evanescent branches (B) at the cut-off/cut-on.*

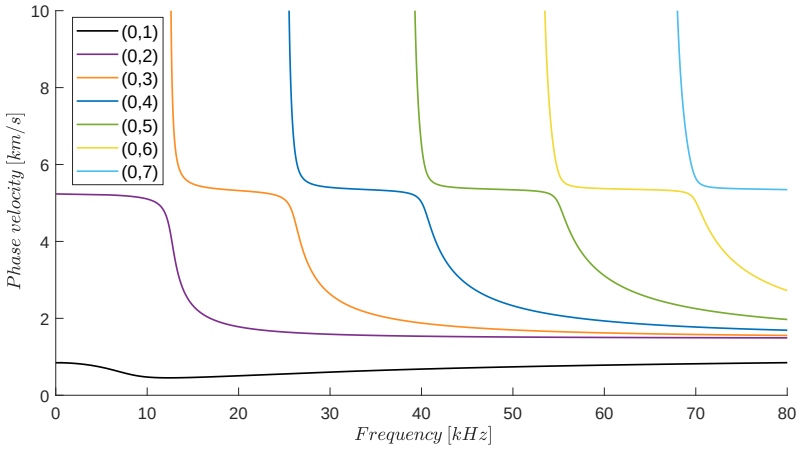


Figure 4.4: *Phase velocity v_p for an aluminium pipe filled with inviscid water. $n = 0$, $W_1 = 49.20mm$, $W_2 = 50.80mm$. v_p is higher than the max speed of the mediums around the cut-on.*

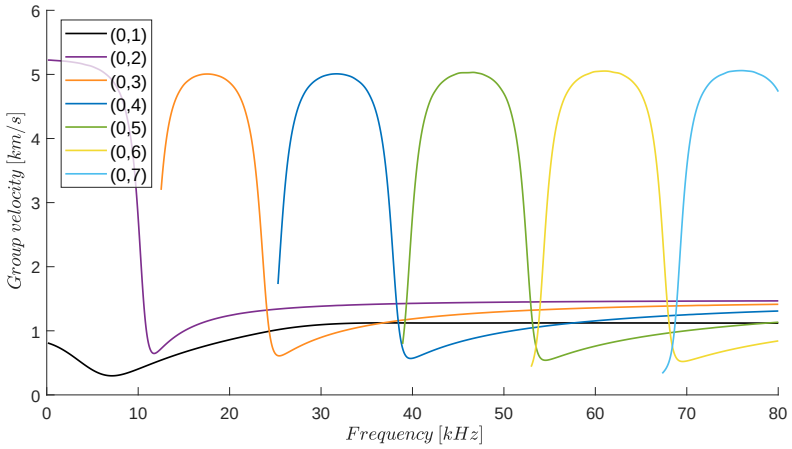


Figure 4.5: *Group velocity v_g for an aluminium pipe filled with inviscid water. $n = 0$, $W_1 = 49.20mm$, $W_2 = 50.80mm$. v_g is null at cut-on and always lower than the max speed of the mediums.*

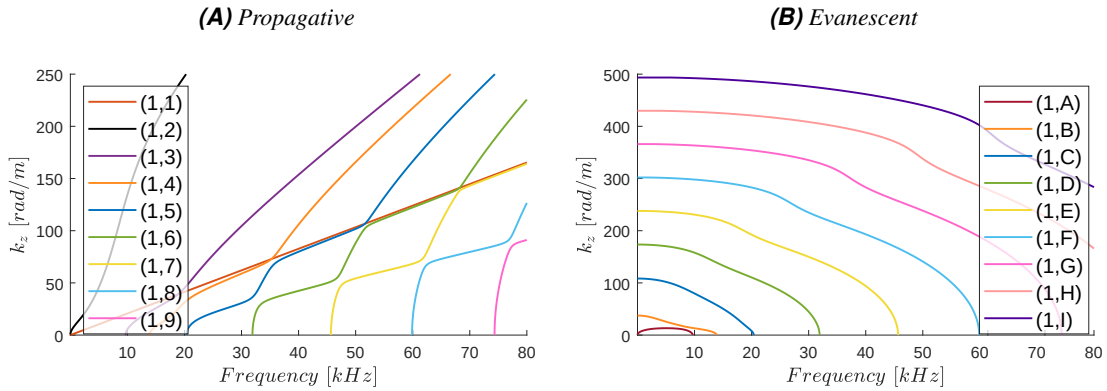


Figure 4.6: *Axial wavenumber $k_{z_{1m}}$ for an aluminium pipe filled with inviscid water. $n = 1$, $W_1 = 49.20mm$, $W_2 = 50.80mm$.*

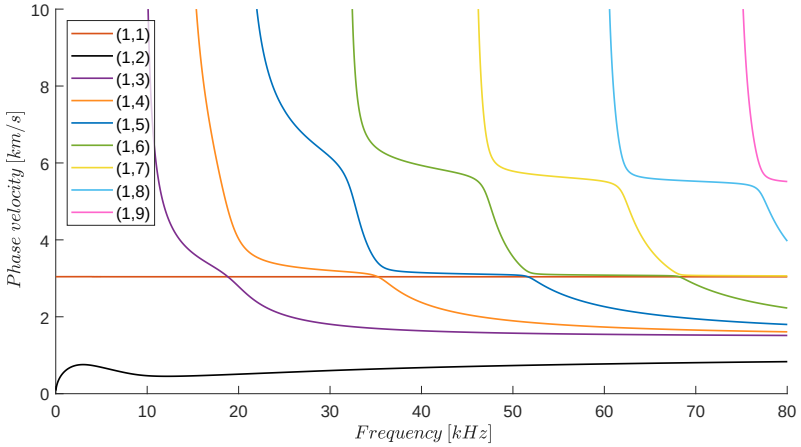


Figure 4.7: *Phase velocity v_p for an aluminium pipe filled with inviscid water. $n = 1$, $W_1 = 49.20mm$, $W_2 = 50.80mm$.*

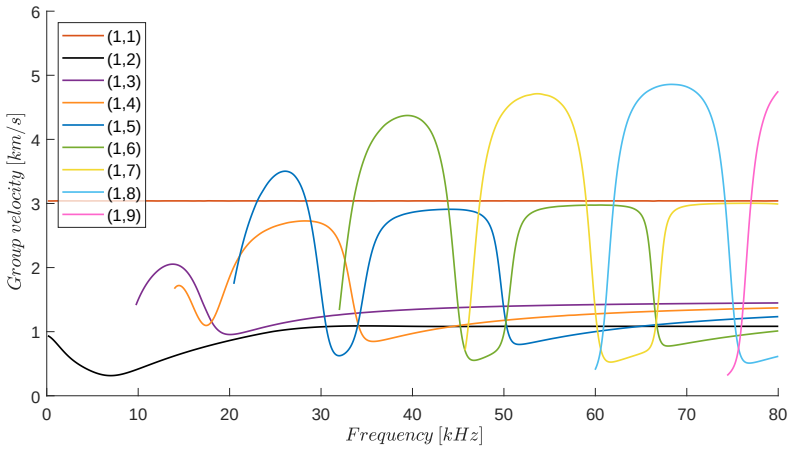


Figure 4.8: *Group velocity v_g for an aluminium pipe filled with inviscid water. $n = 1$, $W_1 = 49.20mm$, $W_2 = 50.80mm$.*

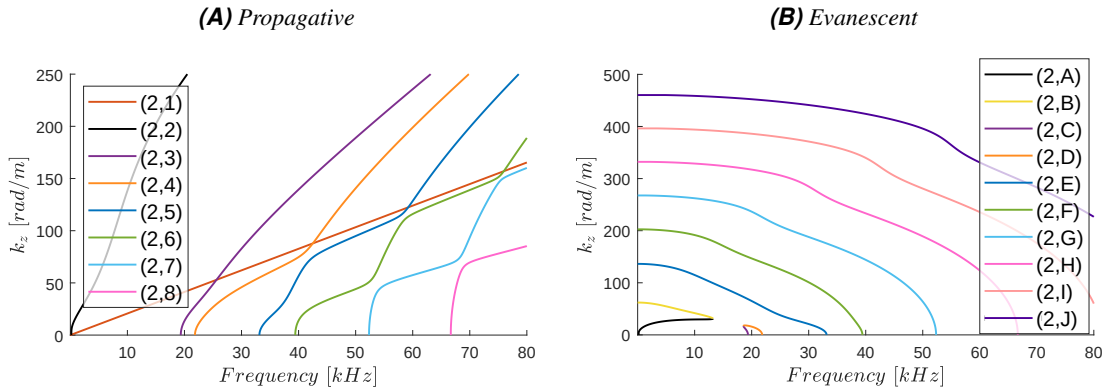


Figure 4.9: *Axial wavenumber $k_{z_{2m}}$ for an aluminium pipe filled with inviscid water. $n = 2$, $W_1 = 49.20mm$, $W_2 = 50.80mm$. Along the spectrum, some branches can convert from evanescent to propagative and vice-versa.*

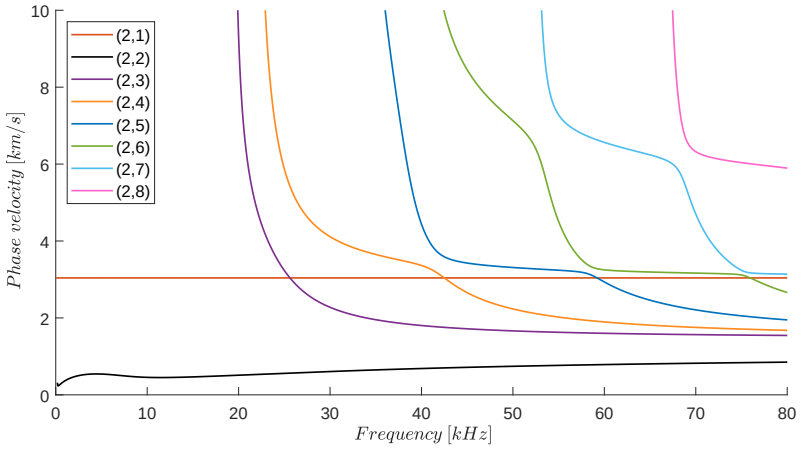


Figure 4.10: *Phase velocity v_p for an aluminium pipe filled with inviscid water. $n = 2$, $W_1 = 49.20mm$, $W_2 = 50.80mm$.*

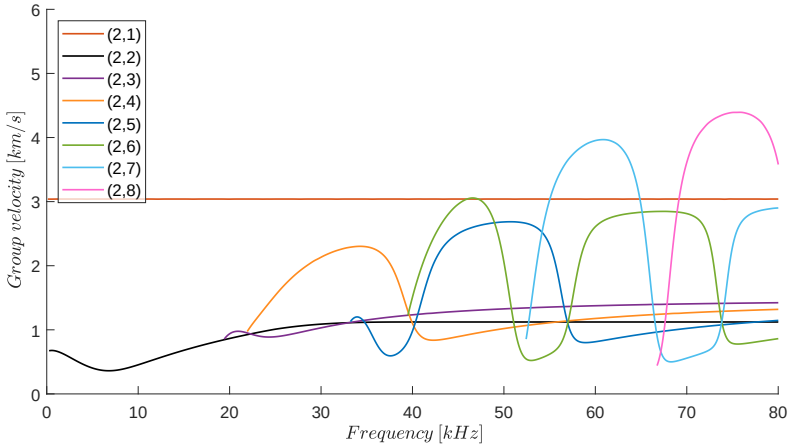


Figure 4.11: *Group velocity v_g for an aluminium pipe filled with inviscid water. $n = 2$, $W_1 = 49.20mm$, $W_2 = 50.80mm$.*

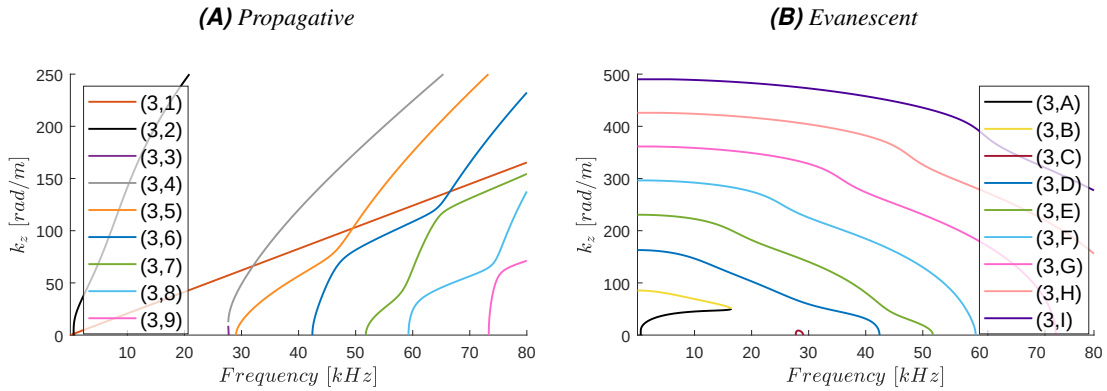


Figure 4.12: *Axial wavenumber k_{z3m} for an aluminium pipe filled with inviscid water. $n = 3$, $W_1 = 49.20mm$, $W_2 = 50.80mm$. Propagative modes $(3,3)$ and $(3,4)$ belong to the same branch but can coexist in a small frequency range.*

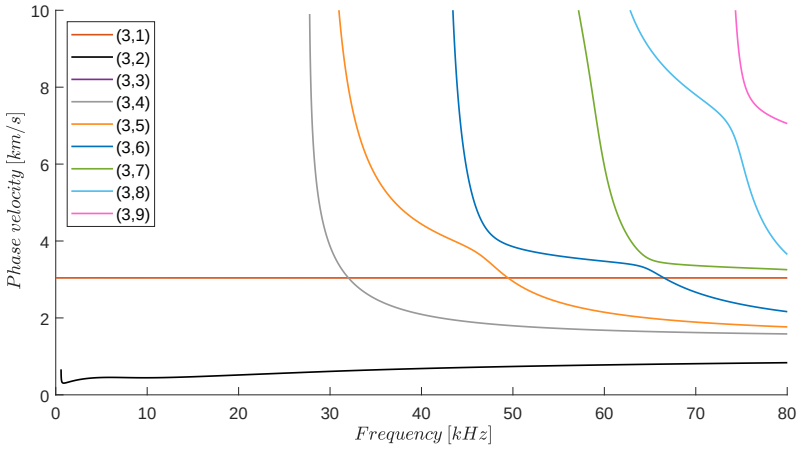


Figure 4.13: *Phase velocity v_p for an aluminium pipe filled with inviscid water. $n = 3$, $W_1 = 49.20mm$, $W_2 = 50.80mm$.*

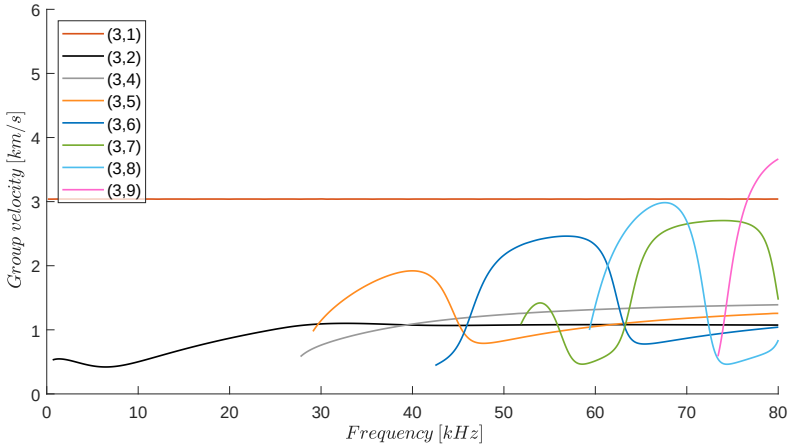


Figure 4.14: *Group velocity v_g for an aluminium pipe filled with inviscid water. $n = 3$, $W_1 = 49.20mm$, $W_2 = 50.80mm$.*

4.1.2 Variation of shell thickness, diameter and material

Wavenumbers, phase velocities, and group velocities clearly depend on the materials and the geometries of the waveguide. This section provides a brief qualitative description of this dependency showing how the results obtained in section 4.1.1.3 are affected by:

- a change in wall thickness,
- a change in pipe diameter,
- a change in pipe material.

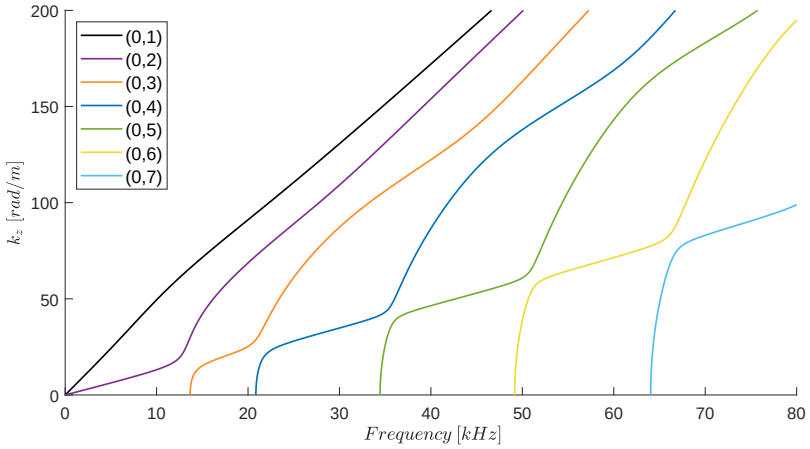


Figure 4.15: *Axial wavenumber $k_{z_{0m}}$ for an aluminium pipe filled with inviscid water. $n = 0$, $W_1 = 44.45mm$, $W_2 = 65.20mm$.*

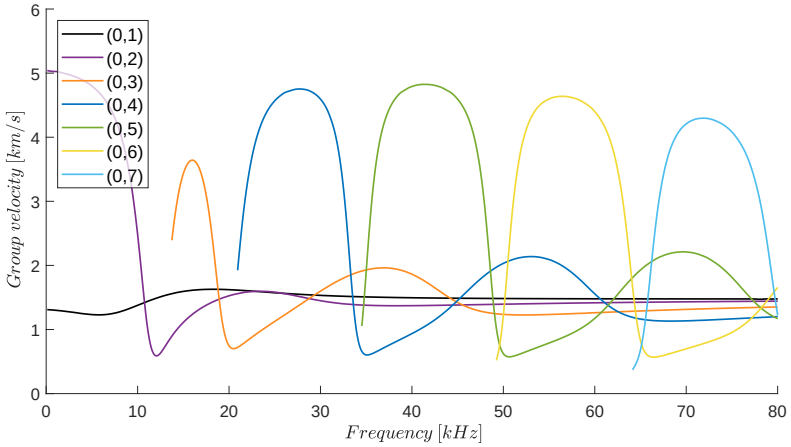


Figure 4.16: *Group velocity v_g for an aluminium pipe filled with inviscid water. $n = 0$, $W_1 = 44.45mm$, $W_2 = 65.20mm$.*

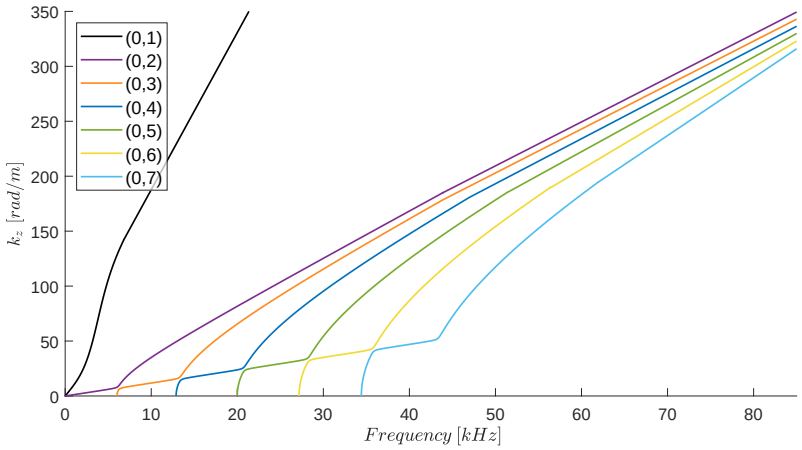


Figure 4.17: *Axial wavenumber $k_{z_{0m}}$ for an aluminium pipe filled with inviscid water. $n = 0$, $W_1 = 98.40mm$, $W_2 = 100mm$.*

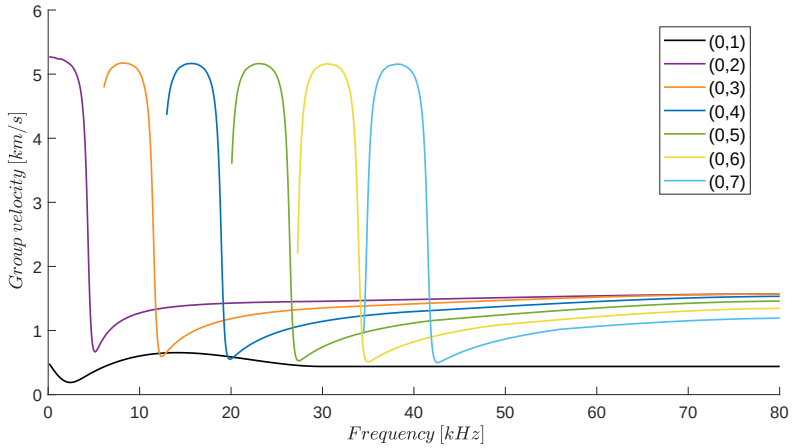


Figure 4.18: *Group velocity v_g for an aluminium pipe filled with inviscid water. $n = 0$, $W_1 = 98.40mm$, $W_2 = 100mm$.*

In the examples reported in this section, only the axisymmetric modes are considered ($n = 0$). Figures 4.15 and 4.16 report, respectively, the axial wavenumber $k_{z_{mn}}$ and the group velocity $v_{g_{nm}}$ for a circular aluminium pipe with inner radius $W_1 = 44.45mm$ and outer radius $W_2 = 65.20mm$. Thus, the wall thickness has been increased by a factor of 10 compared to the case reported in section 4.1.1.3. The shift in the cut-on frequencies is not consistent for all the modes since some of them are translated to lower frequencies (e.g. $(0,4)$, $(0,5)$), while others are translated to higher frequencies (e.g. $(0,3)$). Despite the remarkable change in thickness, the shifts appear to be small, and this seems to be consistent with the

fact that the overall dimensions have not been excessively modified: an acoustic ray travelling along the waveguide runs through similar paths in the two different cases. The change of thickness, however, seems to have a stronger impact on the dispersive behaviour of the waveguide. In the frequency range analysed, despite evident similarities, all the modes exhibit a second overshot before converging to their asymptotic value. Moreover, mode $(0,1)$ propagates faster with a thicker wall.

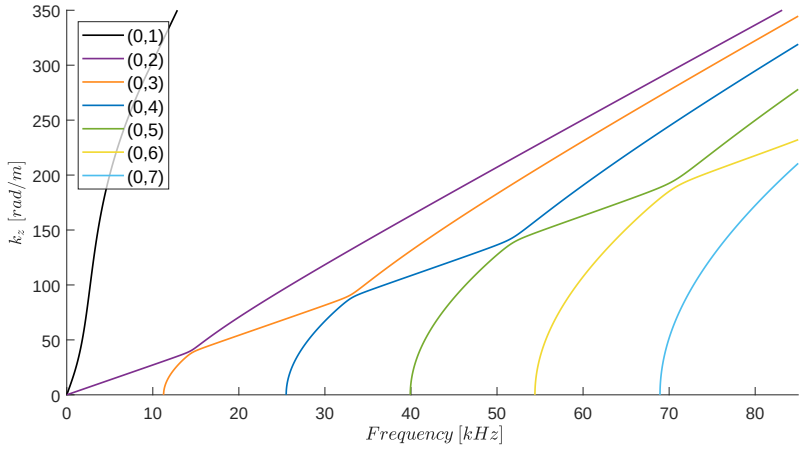


Figure 4.19: *Axial wavenumber $k_{z_{0m}}$ for a PMMA pipe filled with inviscid water. $n = 0$, $W_1 = 49.20mm$, $W_2 = 50.80mm$.*

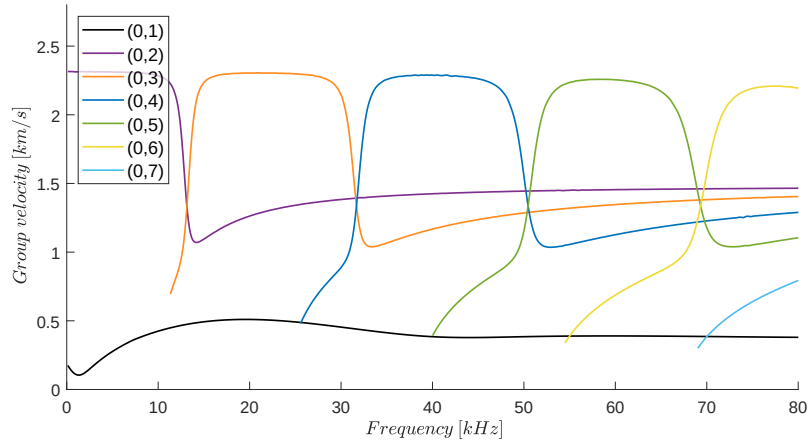


Figure 4.20: *Group velocity v_g for a PMMA pipe filled with inviscid water. $n = 0$, $W_1 = 49.20mm$, $W_2 = 50.80mm$.*

Figures 4.17 and 4.18 report the first seven modes for an aluminium pipe with inner radius $W_1 = 98.40mm$ and outer radius $W_2 = 100mm$. Compared to the case

reported in section 4.1.1.3, the inner radius has been increased by a factor 2 while the wall thickness remains unchanged. Clearly, a larger diameter shifts all the cut-on to lower frequencies, while the dispersive behaviour appears to be marginally affected. This result suggests that dispersive effects in larger pipes with lower frequency signals can be investigated using smaller pipes and signals with a larger frequency range. Consequently, a significant cost reduction could be achieved if a test rig was required. At the same time, it should be noted that higher frequency signals have roughly non-dispersive behaviour for the lower order modes and, for the latter, the propagation develops at a constant group velocity.

The last case reported in figures 4.19 and 4.20 concerns a pipe with the same dimensions as the one in 4.1.1.3 but with a shell made of different material: PMMA. Interestingly, the cut-on frequencies appear almost identical to those for the aluminium case, confirming, once again, that geometry is the main factor in the determination of the frequency range of the modes. The dispersive behaviour, however, is visibly affected by the new material. As for the aluminium case, PMMA longitudinal velocity sets the upper limit for all the group velocities, while modes $(n, 1)$ for $n > 0$ propagates at PMMA shear velocity. Again, almost all the modes asymptotically propagate at the speed of sound in water.

4.2 Techniques for the measurement of the dispersion

This section provides a few techniques for the practical measurements of modal dispersive effects in an actual test rig.

4.2.1 Experimental measurement of group velocity

As seen in section 2.7, the group velocity indicates how fast the energy associated with each mode propagates along the waveguide. A generic signal, in general, spreads its spectrum across several frequencies and stimulates multiple modes. Consequently, different frequency components experience different delays, and a listener at a distance receives a signal whose duration is longer than the original one [73, pp. 161]. If the propagating signal was a train of sine pulses with

a single frequency component at $\bar{\omega}$, all the stimulated modes would propagate with their own specific group velocity $v_{g_{nm}}(\bar{\omega})$, and the receiver would see several trains of pulses separated in time accordingly. Hence, reporting in a *spectrogram* the time domain envelope of the signals at the receiver per each frequency, it is possible to visualise every single mode separately and estimate the numeric value of the group velocity per each mode.

To implement the procedure, the test signal should have a duration Δt short enough to create a clear time gap at the receiver between modes propagating at different speeds. Ideally, it is also necessary to have a band $\Delta\omega$ narrow enough to measure a single value of the group velocity per each mode [80, p. 89]. Therefore, for the requirements above, a reasonable trade-off should be found. Note that increasing the distance between source and receiver Δz loosens the requirements for both Δt and $\Delta\omega$ and, consequently, improves the level of details that can be represented in the spectrogram. However, from a practical point of view, building a test rig using a long pipe might be unfeasible. Therefore, the distance Δz is assumed given, and the test pulse specifications are defined accordingly. Firstly, a simple condition on Δt can be imposed assuming:

$$\Delta t \ll \frac{\Delta z}{max\big(v_{g_{nm}}(\omega)\big)} = T_{min} \qquad \forall\omega, \tag{4.5}$$

where T_{min} is the minimum propagation time between the source and the receiver. Condition (4.5) aims to minimise the overlapping between the train of pulses associated with different modes. The condition on $\Delta\omega$ can be found imposing a low time spread of the pulses received. Assuming a narrow band pulse with frequencies between $\bar{\omega}$ and ω and a variation of the group velocity approximately linear in the interval, a short pulse spread at the receiver means:

$$\frac{\Delta z}{v_{g_{nm}}(\omega)} - \frac{\Delta z}{v_{g_{nm}}(\bar{\omega})} \ll \Delta t. \tag{4.6}$$

According to the condition above, a test pulse has a low time spread when the difference between the arrival time of its components at max and min frequencies is

much smaller than the duration of the pulse itself. Hence, since $v_{g_{nm}}(\omega) \approx v_{g_{nm}}(\bar{\omega})$, $v_{g_{nm}}(\omega)$ can be expanded using the Taylor series in $v_{g_{nm}}(\bar{\omega})$ and (4.6) yields:

$$v_{g_{nm}}(\bar{\omega}) - \left[v_{g_{nm}}(\bar{\omega}) + \frac{\partial v_{g_{nm}}(\omega)}{\partial \omega}\Big|_{\bar{\omega}} (\omega - \bar{\omega}) \right] \ll \frac{\Delta t}{\Delta z} v_{g_{nm}}(\omega) v_{g_{nm}}(\bar{\omega}), \tag{4.7}$$

which provides the condition for the frequency boundaries of the pulse:

$$\Delta\omega = \omega - \bar{\omega} \ll \frac{\Delta t}{\Delta z} \frac{v_{g_{nm}}(\omega)}{v_{g_{nm}}(\bar{\omega})} \left(\frac{\partial^2 k_{z_{nm}}}{\partial \omega^2}\Big|_{\bar{\omega}} \right)^{-1} \approx \frac{\Delta t}{\Delta z} \left(\frac{\partial^2 k_{z_{nm}}}{\partial \omega^2}\Big|_{\bar{\omega}} \right)^{-1} = \frac{\Delta t}{\Delta z} d_{nm}^{-1}(\bar{\omega}), \tag{4.8}$$

where $d_{nm}(\bar{\omega})$ is called *pulse spreading* of mode nm at frequency $\bar{\omega}$. Since a single line of the spectrogram includes all modes at a given frequency, $\Delta\omega$ should be determined, where possible, using the maximum value of $d_{nm}(\omega)$ among all the propagative modes. Hence:

$$\Delta\omega \ll \frac{\Delta t}{\Delta z \; max\, d_{nm}} \quad \forall \omega \;\; with \;\; \Delta t \ll T_{min}. \tag{4.9}$$

The spectrogram is built by dividing the frequency spectrum into intervals $[\bar{\omega}, \bar{\omega} + \Delta\omega]$, with $\bar{\omega} = n\Delta\omega$, and stacking together the recordings of the envelopes of the related narrow band pulses in the time domain. Indeed, the pressure signal at the receiver can be thought of as a sum of several pressure signals all shifted by a specific delay given by the related group velocity. This can be shown by considering a narrow band pressure signal of central frequency $\bar{\omega}$:

$$\hat{p}_{\bar{\omega}}(r,\theta,z,\omega) = \rho\omega^2 \hat{\phi}_{\bar{\omega}}(r,\theta,z,\omega) = \rho\omega^2 \sum_{n=-\infty}^{+\infty} \sum_{m=1}^{+\infty} \hat{w}\left(\frac{\omega - \bar{\omega}}{\Delta\omega} \right) \Phi_{nm}(\omega) J_n(q_{\phi_{nm}}(\omega) r) e^{in\theta} e^{ik_{znm}(\omega) z}, \tag{4.10}$$

where the $\Delta\omega$ narrow band is explicitly introduced by using $\hat{w}(\omega)$, a window function with unitary bandwidth. Since the test signal is narrow band, the axial wavenumber $k_{z_{nm}}(\omega)$ can be expanded in Taylor series using only the first

derivative:

$$k_{z_{nm}}(\omega) \approx k_{z_{nm}}(\bar{\omega}) + \frac{\partial k_{z_{nm}}}{\partial \omega}\bigg|_{\bar{\omega}} (\omega - \bar{\omega}) = k_{z_{nm}}(\bar{\omega}) + \frac{\omega - \bar{\omega}}{v_{g_{nm}}(\bar{\omega})}. \tag{4.11}$$

Plugging (4.11) into (4.10) yields:

$$\begin{aligned} &\hat{p}_{\bar{\omega}}(r,\theta,z,\omega) = \\ &\sum_{n=-\infty}^{+\infty} \sum_{m=1}^{+\infty} \rho\omega^2 \Phi_{nm}(\omega) J_n(q_{\phi_{nm}}(\omega) r) e^{in\theta} e^{i(k_{znm}(\bar{\omega})z - \bar{\omega}z/v_{gnm}(\bar{\omega}))} \\ &\qquad \cdot \hat{w}\Big(\frac{\omega - \bar{\omega}}{\Delta\omega}\Big) e^{i\omega z/v_{gnm}(\bar{\omega})}. \end{aligned} \tag{4.12}$$

Assuming Δz as the position of the receiver, the quantity $\tau_{nm}(\bar{\omega}) = \Delta z / v_{g_{nm}}(\bar{\omega})$ represents the time required for the mode nm to propagate from the source to the receiver at frequency $\bar{\omega}$. Applying the inverse Fourier transform with respect to ω to the previous equation yields:

$$\begin{aligned} &p_{\bar{\omega}}(r,\theta,\Delta z,t) = \\ &\sum_{n=-\infty}^{+\infty} \sum_{m=1}^{+\infty} h_{nm}(r,\theta,\Delta z,t) \star e^{i\bar{\omega}t} w(t\Delta\omega)\Delta\omega \star \delta(t - \tau_{nm}(\bar{\omega})) = \\ &\Delta\omega \sum_{n=-\infty}^{+\infty} \sum_{m=1}^{+\infty} h_{nm}(r,\theta,\Delta z,t) \star w\big((t - \tau_{nm}(\bar{\omega}))\Delta\omega\big) e^{i\bar{\omega}(t - \tau_{nm}(\bar{\omega}))}, \end{aligned} \tag{4.13}$$

where $\star$ denotes the convolution, $w(t)$ is the inverse Fourier transform of the window function $\hat{w}(\omega)$, and $h_{nm}(r,\theta,z,t)$ is the inverse Fourier transform of:

$$\hat{h}_{nm}(r,\theta,z,\omega) = \rho\omega^2 \Phi_{nm}(\omega) J_n(q_{\phi_{nm}}(\omega) r) e^{in\theta} e^{i(k_{znm}(\bar{\omega})z - \bar{\omega}z/v_{gnm}(\bar{\omega}))}. \tag{4.14}$$

Equation (4.13) describes the pressure at the receiver as a sum of modes, each delayed by its own group velocity at frequency $\bar{\omega}$. Consequently, the group velocities can be determined by measuring the delays of the received signals in the time domain and organising them in a spectrogram. A practical application of the method described here is reported in section 4.4.

4.2.2 Experimental measurement of modal dispersion

The modal dispersion curves calculated in section 4.1 can be measured empirically to validate the model. A similar procedure has already been described in several works [89],[67] and it is here reported to provide complementary measurements to those for the group velocity of the previous section. For the inner liquid, the pressure at frequency ω can be written as (equation 4.1):

$$\hat{p}(r,\theta,z,\omega) = \rho\omega^2\hat{\phi}(r,\theta,z,\omega) = \rho\omega^2 \sum_{n=-\infty}^{+\infty}\sum_{m=1}^{+\infty} \Phi_{nm} J_n(q_{\phi_{nm}} r) e^{in\theta} e^{ik_{z_{nm}} z}. \quad (4.15)$$

For a given ω, the equation above can be space Fourier transformed along z as follows:

$$\begin{aligned}
\hat{p}(r,\theta,\bar{k}_z,\omega) &= \frac{\rho\omega^2}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty}\sum_{m=1}^{+\infty} \Phi_{nm} J_n(q_{\phi_{nm}} r) e^{in\theta} e^{ik_{z_{nm}} z} e^{-i\bar{k}_z z} dz \\
&= \sum_{n=-\infty}^{+\infty}\sum_{m=1}^{+\infty} \frac{\rho\omega^2}{\sqrt{2\pi}} \Phi_{nm} e^{in\theta} J_n(q_{\phi_{nm}} r) \int_{-\infty}^{+\infty} e^{ik_{z_{nm}} z} e^{-i\bar{k}_z z} dz \\
&= \sum_{n=-\infty}^{+\infty}\sum_{m=1}^{+\infty} \frac{\rho\omega^2}{\sqrt{2\pi}} \Phi_{nm} e^{in\theta} J_n(q_{\phi_{nm}} r) \delta(\bar{k}_z - k_{z_{nm}}).
\end{aligned} \quad (4.16)$$

Equation (4.16) shows that, when $\hat{p}(r,\theta,\bar{k}_z,\omega)$ is represented in the $\omega - \bar{k}_z$ space, the space transformed pressure amplitude (thus its energy) exhibits non-null values only for $\bar{k}_z = k_{z_{nm}}$. Therefore, in the $\omega - \bar{k}_z$ plane, modes are identified by those points where the pressure amplitude is not null. Note that for $\bar{k}_z = k_{z_{nm}}$ the values also depend on the modal amplitude. To build the representation in the $\omega - k_z$ plane, per each value of ω in the desired frequency range, it is necessary to measure the value of the pressure amplitude along the pipe, keeping constant the r and θ coordinates of the receiver. Once the pressure along z is known, it can be Fourier transformed to get the representation in k_z. At a given frequency, a complete measurement along z provides a single line in the $\omega - k_z$ plane. The whole representation is obtained by repeating the measurement in the desired frequency range. This method provides fairly accurate results, but it is obviously

tedious, and care should be taken when the measurements are performed.

As reported in [89] and [67], the procedure is generally implemented using a pressure source and a hydrophone that is moved along the pipe. It is important to remark that a source placed in a non-axisymmetric position stimulates modes for several values of n simultaneously, while one centred on the axis stimulates only modes for $n = 0$. Furthermore, to include modes for $n \neq 0$ in the recordings, the hydrophone must be placed away from the axis. Since $n \neq 0$ modes cannot be measured separately as can be done for $n = 0$, their exact determination on the $\omega - k_z$ plane can be arduous. A possible workaround consists in repeating the measurement with different positions of the hydrophone around the axis to exploit the $e^{in\theta}$ dependence of the modes. Other techniques to achieve a certain degree of isolation for different modes are mentioned in [68, p. 202]. In theory, the test signal used should be a pure sine with a single pulse in its single-sided frequency spectrum. However, since a real test rig has finite dimensions, a better option is a train of sine pulses a few periods long. The duration, in fact, must be determined as a trade-off between keeping the signal band as narrow as possible (which is achieved with a longer duration) and the necessity to avoid disturbances introduced by signals bounced back from the pipe terminals (which is achieved by stopping the recording before the arrival of the first reflected signal at the hydrophone) [80, p. 89]. A further observation concerns the resolution of the steps along the z axis, which should be set according to the Nyquist theorem with respect to the maximum wavenumber range associated with k_z. For instance, if the k_z spectrum ranges from 0 to $250 rad/m$, a z sampling step of no more than $12mm$ would guarantee a correct application of the Nyquist theorem for the space Fourier transform. On the other hand, the frequency step depends on the desired resolution. For instance, assuming that the signal to reverberate is $100ms$ long and sampled at $100kHz$, if one measured pressure value is desired per each signal sample, a frequency step of $10Hz$ is required.

A second method is also mentioned in [89], where the dispersion is calculated by determining the phase velocity from the phase difference between two fixed points along the pipe. Although much simpler in terms of the number of measurements

required, the final accuracy is lower. A slightly different approach is also mentioned in [67], where $k_{z_{nm}}$ is determined from the group velocity measured at a single point. This approach is similar to the one explained in section 4.2.1, where the modal group velocity $v_{g_{nm}}$ is extrapolated from the spectrogram. The axial wavenumber $k_{z_{nm}}$ is then assessed directly from group velocity definition as:

$$k_{z_{mn}}(\omega) = \int_0^{\omega} \frac{1}{v_{g_{nm}}(\omega)} d\omega + k_{z_{mn}}(0), \tag{4.17}$$

where the integration constant $k_{z_{mn}}(0)$ can be generally assumed null.

4.3 Modelling the pressure source

This section describes the second layer of the acoustic model, which accounts for the integration of the acoustic sources. In this work, the acoustic source S_P is assumed to be placed in the inner liquid domain on a plane orthogonal to the axis of the waveguide (figure 4.1). For convenience, the plane is chosen to be $z = 0$. Although it is possible to account for different shapes, the source is also assumed to be circular.

The integration of the source into the mathematical model can be achieved by finding the amplitude of the modal potentials so that the pressure distribution of the source is correctly reconstructed by the sum of the modes at the source position. Nevertheless, this problem exhibits at least two important differences compared to the case reported in section 2.8. Indeed, the simple free boundary condition of a liquid cylinder offers a modal decomposition where the eigenfunctions are orthogonal and form a complete basis for the space of the potentials. The orthogonality allows modal decoupling and a precise calculation of the modal amplitudes, while the completeness guarantees that every potential function can be represented by a certain combination of modes. As seen in section 2.1.2.1 and 4.1.1.2, the liquid-solid interface invalidates these hypotheses, meaning that orthogonality and completeness can no longer be assumed verified [138],[102, p. 363, 374]. Because of the lack of orthogonality, the modal amplitudes cannot

be determined independently. For a given order n, modes (n,x) are coupled together, and, in theory, an infinite number of modes should be accounted for when determining the amplitudes. Fortunately, a reasonable truncation allows a good approximation of the original pressure distribution, and the amplitude of the potentials can be determined considering only a finite number of modes. The potential lack of completeness means that some potential functions might not be written as a combination of the modes found from the homogeneous equation. Although completeness is important for a rigorous approach, in the literature, it has been assumed true [105],[106] or not explicitly mentioned [107], and this seems to be a problem still unsolved[2]. In the following, it is also assumed verified.

Two different novel methods are proposed for calculating the coupled amplitudes, and they both rely on the formulation of the pressure field emitted by a source in a half-space. Other approaches based on the formulation of the Green function [235],[236] are theoretically possible but, as explained in section 2.1.2.1, not convenient when orthogonality is missing. Since the model is meant to be coupled with a simple test rig, the source here considered is a *baffled circular piston*, which is a good trade-off between complexity and accuracy when the emitter is an electrodynamic speaker. The same procedures, however, can be used for any pressure distribution in the plane $z=0$ [3]. As shown, the only concern relates to the number of modes to include in the calculation: the more intricate the pressure distribution, the higher the number of modes that should be considered.

To the best of our knowledge, the proposed methods are new. However, other examples of source integration certainly already exist in the literature. Some issues reported in [105] by Alonso have been considered to develop the methods used in this work. A range of other examples concerning source modelling in

[2] Completeness, however, cannot be assumed true only for rigid and free boundary conditions. For example, considering a simple cylinder, if boundary conditions are given as a function of the impedance Z, free and rigid boundaries are obtained for $Z=0$ and $Z=\infty$, respectively. If we assume that for $Z\neq 0$ and $Z\neq\infty$ the set of modes is not complete, there must be at least a function f that cannot be written as a linear combination of modes for any value of Z different from 0 or ∞. However, since modes can be thought of as a continuous function of the impedance Z, their summation in $Z=0$ or $Z=\infty$ must be equal to the summation on the limit for $Z\to 0$ or $Z\to\infty$, respectively. If a certain summation for $Z=0$ or $Z=\infty$ reconstructed the given function f, assuming the completeness only in $Z=0$ and $Z=\infty$ gives a contradiction since there would be a summation of modes that, for $Z\to 0$ or $Z\to\infty$, converges at the same time to f and to a different function.

[3] Used in combination with FEM techniques, the model proposed could be extended to sources with arbitrary distribution by calculating the equivalent pressure in $z=0$.

different conditions can also be found. For instance, Baik [76] describes the source modelling when the source creates a cross-sectional excitation. Alonso [106] and Rienstra [107] provide a solution to model the source when the coupling between the inner fluid and the outer shell is known in terms of wall impedance.

4.3.1 Decomposition of a baffled pressure source

Speakers are generally modelled as *circular pistons* [122, p. 459], and datasheets usually provide emission diagrams, sound pressure levels and other electro-acoustic data. These specifications, however, can only be partially predicted by the circular piston model since they also depend on other specific electro-mechanical features [65, p. 406],[237]. Besides, specifications are medium dependant, and a new characterisation is required if the device is used in a different medium[4]. Here, as a reasonable trade-off between complexity and accuracy, the source is calculated as a circular piston in inviscid water.

A model for a source of finite extension, such as a piston, can be obtained from the pressure field of a monopole that pulsates with velocity $\vec{\boldsymbol{v}}$ in an inviscid fluid medium [239, p. 107]. Its harmonic pressure field can be written as:

$$p(d,t) = -\frac{i\rho cQ}{2\lambda d} e^{i(kd-\omega t)}, \tag{4.18}$$

where d is the distance from the source, ρ is the density of the medium, c is the intrinsic acoustic velocity of the medium, and λ is the wavelength at the given frequency. The *source strength* Q is:

$$Qe^{-i\omega t} = \int_S \vec{\boldsymbol{v}} \cdot \vec{\boldsymbol{n}} dS, \tag{4.19}$$

with the integral of the normal velocity $\vec{\boldsymbol{v}}$ calculated on the surface S of the source. Equation (4.18) is valid when the dimension of the source is much smaller than λ. When the source is placed on a rigid plane boundary - usually named *baffle* -,

[4] It is not always possible to employ common speakers for underwater acoustic. However, there exist simple devices that are waterproof and can be used as a simple alternative to more costly and more adequate emitters. For instance, [238] is an inexpensive waterproof speaker that can be submerged without incurring any damage. The dimensions of this device are used as a reference in the following sections.

using the *method of images* [65, p. 163], it is possible to demonstrate that the pressure amplitude doubles. Therefore, the pressure of a *baffled monopole* is:

$$p(d,t) = -\frac{i\rho cQ}{\lambda d} e^{i(kd-\omega t)}. \tag{4.20}$$

When a source has finite dimensions, the pressure field can be found by decomposing the source into infinitesimal components. All the contributions are then summed up to obtain the desired finite extension. Hence, considering a baffled flat piston of radius R_P whose surface moves in the direction of the symmetry axis with velocity $\hat{v}(\omega)e^{i\omega t}$ (figure 4.21), the pressure field at the point $G \equiv (r,\theta,z)$ in the frequency domain can be calculated as:

$$\hat{p}(G,\omega) = -\frac{i\rho c}{\lambda}\int_{Q_P}\frac{e^{ikd}}{d}dQ = -\frac{i\rho c\hat{v}}{\lambda}\int_{S_P}\frac{e^{ikd}}{d}dS_P, \tag{4.21}$$

where S_P is the surface of the piston, dS_P its infinitesimal part, and d the euclidean distance between G and dS_P.

Decomposing large radiating surfaces into small radiating elements is also a common approach in FEM [68, p. 214]. Arbitrary sources are generally decomposed as a collection of monopoles or bipoles, and, to avoid singularities, no pressure value is given at the exact position of the simple sources. The same issue also exists for 4.21.

4.3.2 Pressure field of a circular baffled piston

The pressure field given by (4.21) can be calculated either numerically or analytically. In the latter case, however, the solution can only be found in the far-field approximation or along the symmetry axis. The far-field representation provides a general idea about the behaviour of the source while the numeric calculation is used to find the pressure in the proximity of the radiating surface. To calculate the far-field radiation, it is convenient to adopt a spherical reference system as shown in figure 4.21.

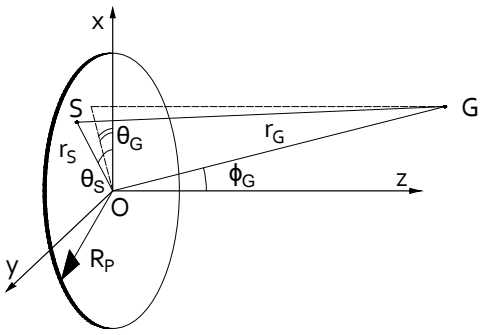


Figure 4.21: *Spherical reference system for the radiating piston.*

Assuming:

$$x = r\sin\phi\cos\theta \tag{4.22a}$$

$$y = r\sin\phi\sin\theta \tag{4.22b}$$

$$z = r\cos\phi, \tag{4.22c}$$

the euclidean distance between two points S and G is:

$$d(G,S) = \sqrt{(x_G - x_S)^2 + (y_G - y_S)^2 + (z_G - z_S)^2} = \tag{4.23}$$

$$\sqrt{r_G^2 + r_S^2 - 2r_G r_S \alpha(\phi_G, \theta_G, \phi_S, \theta_S)}, \tag{4.24}$$

where:

$$\alpha(\phi_G, \theta_G, \phi_S, \theta_S) = \tag{4.25}$$

$$\sin\phi_G\cos\theta_G\sin\phi_S\cos\theta_S + \sin\phi_G\sin\theta_G\sin\phi_S\sin\theta_S + \cos\phi_G\cos\phi_S. \tag{4.26}$$

In the far-field approximation, the dimension of the source is negligible in respect of the distance between the source and the observation point. The same distance is also much bigger than the wavelength, thus:

$$d(G,S) \approx d(G,O) = r_G \quad \forall S \in S_P \tag{4.27a}$$

$$d(G,O) \gg \lambda, \tag{4.27b}$$

where O is the centre of the piston and the origin of the reference system. Approximations (4.27) can be used to simplify the integration of (4.21). The denominator of the integrand function:

$$\frac{e^{ikd(G,S)}}{d(G,S)} \tag{4.28}$$

can be simply replaced by r_G since this substitution only introduces a little inaccuracy on the amplitude. The numerator carries the phase of a complex exponential, and the angle dependencies cannot be neglected. Hence:

$$d(G,S) = \sqrt{r_G^2 + r_S^2 - 2r_G r_S \alpha} \approx r_G \sqrt{1 - 2\frac{r_S}{r_G}\alpha} \approx r_G\left(1 - \frac{r_S}{r_G}\alpha\right), \tag{4.29}$$

where the previous relationship is found for $r_S/r_G \ll 1$ and from the Taylor expansion up to the first derivative of the function $\sqrt{1-2\frac{r_S}{r_G}\alpha}$. With the previous two approximations, equation (4.21) can be rewritten as:

$$\hat{p}(G,\omega) = -\frac{i\rho c\hat{v}}{\lambda}\int_{S_P}\frac{e^{ikd}}{d}dS_P \approx \\ -i\rho c\hat{v}\,\frac{e^{ikr_G}}{\lambda r_G}\int_{S_P} e^{-ikr_S\alpha(\phi_G,\theta_G,\phi_S,\theta_S)}dS_P. \tag{4.30}$$

Now, considering that the integration is on the surface of the piston, where $\phi_S = \pi/2$:

$$\hat{p}(G,\omega) \approx -i\rho c\hat{v}\,\frac{e^{ikr_G}}{\lambda r_G}\int_{S_P} e^{-ikr_S\sin\phi_G\cos(\theta_G-\theta_S)}dS_P. \tag{4.31}$$

Hence:

$$\hat{p}(G,\omega) \approx -i\rho c\hat{v}\,\frac{e^{ikr_G}}{\lambda r_G}\int_0^{R_P} r_S \int_0^{2\pi} e^{-ikr_S \sin\phi_G \cos\theta_S} d\theta_S dr_S = \\ -i\rho c\hat{v}\,\frac{e^{ikr_G}}{\lambda r_G}\int_0^{R_P} r_S 2\pi J_0(kr_S \sin\phi_G) dr_S, \tag{4.32}$$

where the integral form of the Bessel function:

$$J_0(x) = \frac{1}{2\pi}\int_0^{2\pi} e^{-ix\cos\theta} d\theta \tag{4.33}$$

has been used. Changing the integration variable with $z = r_S k \sin\phi_G$ yields:

$$\hat{p}(G,\omega) \approx -i\rho c\hat{v}\,\frac{e^{ikr_G}}{\lambda r_G}\frac{2\pi}{k^2\sin^2\phi_G}\int_0^{R_P k\sin\phi_G} z J_0(z) dz = \tag{4.34}$$
$$-i\rho c\hat{v}\,\frac{e^{ikr_G}}{\lambda r_G}\frac{2\pi R_P^2 J_1(kR_P\sin\phi_G)}{kR_P\sin\phi_G} = -i\omega\rho\hat{v}\,R_P^2\frac{e^{ikr_G}}{r_G}\frac{J_1(kR_P\sin\phi_G)}{kR_P\sin\phi_G},$$

where the following identity has been used:

$$xJ_1(x) = \int_0^x yJ_0(y)dy. \tag{4.35}$$

As expected from the symmetry of the problem, equation (4.34) depends only on the distance from the source r_G and on the angle ϕ_G. Besides, since r_G and ϕ_G appear as independent factors, their contribution can be represented separately. The function:

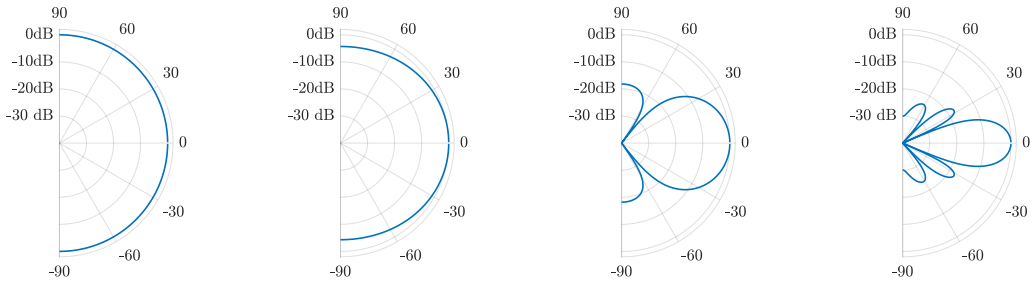
$$B(\phi_G)_{dB} = 20\log\left|\frac{2J_1(kR_P\sin\phi_G)}{kR_P\sin\phi_G}\right| \tag{4.36}$$

can be used to represent the angular dependence in dB scale (figure 4.22). It exhibits a main lobe around $\phi_G = 0$, where it reaches its maximum value $0dB$. Then, depending on the value of kR_P, other secondary lobes can appear on the side of the main one. The pressure nodes that separate different lobes can be found for those values of ϕ_G where:

$$J_1(kR_P \sin\phi_G) = 0. \tag{4.37}$$

Note that when kR_P is smaller than 3.831 (the first zero of $J_1(x)$), only the main lobe is present. Figure 4.22 shows the polar diagram for A) $kR_P = 0.09$, B) $kR_P = 1.91$, C) $kR_P = 4.78$, and D) $kR_P = 9.55$. For very low kR_P (figure 4.22-A), the amplitude of the main lobe tends to be constant for any ϕ_G, and the radiation of a baffled piston resembles the radiation of a baffled monopole. This result is sometimes adopted in FEM simulations [122, p. 45] to simplify the model. For high kR_P (figure 4.22-D), the piston tends to concentrate the radiation along the symmetry axis (high *directivity*), and the approximation with a simple monopole is no longer accurate.

Figure 4.23 shows the radiation from a circular piston in water for the same cases illustrated in figure 4.22. Pressure is given in the scaled dimensionless form $|\hat{p}|/|\rho c\hat{v}|$. Each sub-figure reports the radiation on the rz plane (left side): the beam patterns shown in figure 4.22 can be easily recognised. For the same velocity amplitude, the pressure (and the related intensity radiated) increases with frequency. As for simple sources, pistons are not good radiators when the wavelength is much larger than the radius R_P. The right-hand side of subfigures 4.23 reports the pressure distribution on a plane parallel and close to the surface of the piston at $z = d_m$. The distance d_m is chosen to avoid the singularity given by a null distance between source S and observation point G, when the latter is on the plane of the piston.



(A) *1 kHz* ***(B)*** *20 kHz* ***(C)*** *50 kHz* ***(D)*** *100 kHz*

Figure 4.22: *Beam pattern $B(\phi_G)_{dB}$ of a circular plane piston in water. $R_P = 0.0225mm$, $c = 1480m/s$.*

Note that, when approximating the pressure on the piston surface at high frequency, the distance d_m must be chosen more carefully. Indeed, at high frequency, the pressure exhibits more irregular variations along z (at low frequency, the variation is just $1/r$ as for the monopole).

Some limitations must be remarked for the model proposed in this section. The first issue concerns the movement of the piston surface. In real transducers, the surface vibrates, changing its position. This displacement, however, is neglected in the proposed model. Moreover, the velocity cannot be assumed to be the same at each point of the radiating surface. Speakers, for instance, are actuated by a central coil, and the velocity is higher at the centre and lower toward the rim. Finally, the baffle is considered rigid and infinite, a condition certainly not completely verified in practice. The scattering introduced by the edge of the baffle, for example, has been analysed in a more accurate model by Backman [237]. Despite all these limitations, the model introduced here offers a good trade-off between complexity and accuracy, and it is used in the following sections.

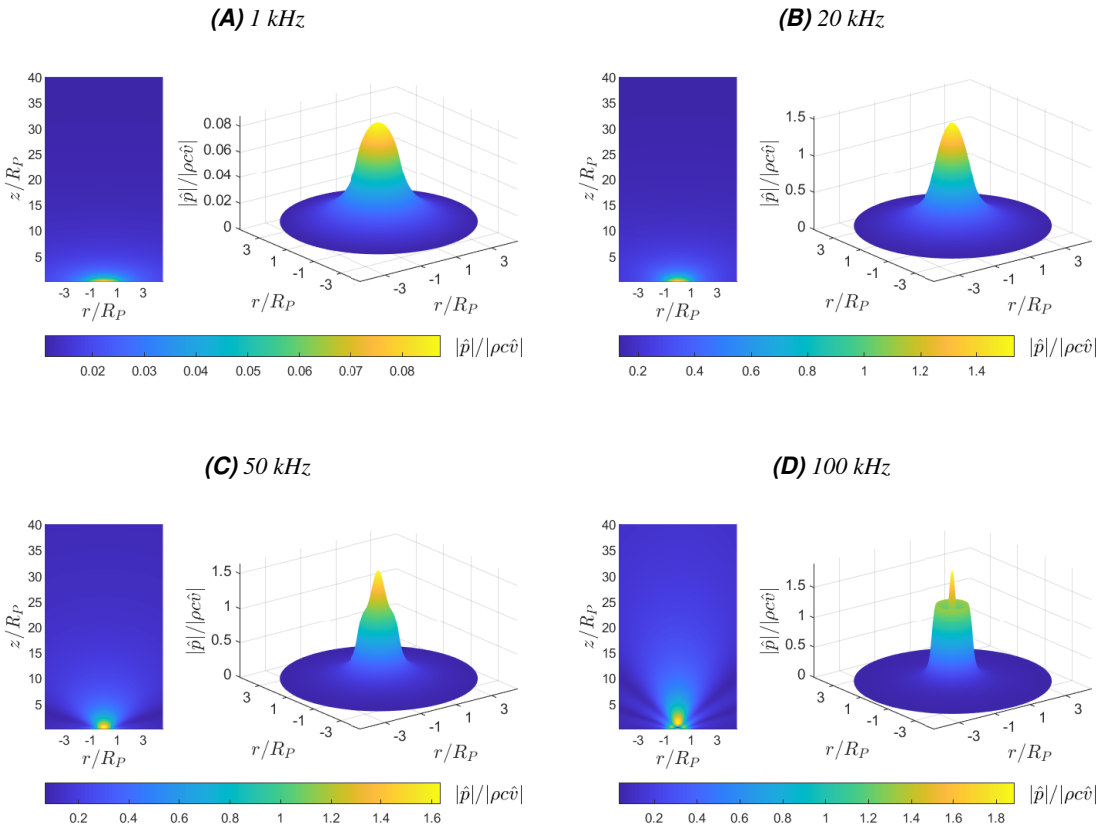


Figure 4.23: *Radiation from a baffled circular piston.* $R_P = 0.0225mm$, $c = 1480m/s$, $d_m = 1mm$.

4.3.3 Integration of the source by distance minimisation

Under the hypothesis of completeness, pressure in the inner liquid domain can be written as (section 2.9.1):

$$\hat{p}(r,\theta,z,\omega) = \rho\omega^2 \sum_{n=-\infty}^{+\infty} \sum_{m=1}^{+\infty} \Phi_{nm} J_n(q_{\phi_{nm}} r) e^{in\theta} e^{ik_{z_{nm}} z}. \tag{4.38}$$

Hence, the pressure distribution in $z = 0$ $\hat{p}_P$ can be written as:

$$\hat{p}_P(r,\theta,\omega) = \sum_{n=-\infty}^{+\infty} \hat{p}_n(r,\theta,\omega) = \rho\omega^2 \sum_{n=-\infty}^{+\infty} \sum_{m=1}^{+\infty} \Phi_{nm} J_n(q_{\phi_{nm}} r) e^{in\theta}, \tag{4.39}$$

where $\hat{p}_n$ is the pressure component related to all the modes with the same angular dependence. Under the assumption of completeness, the set of functions:

$$B = \{\beta_{mn}(r,\theta,\omega)\} = \{J_n(q_{\phi_{nm}} r) e^{in\theta}\} \tag{4.40}$$

form, per each ω, a non-orthogonal basis in the domain $r \in [0, W_1]$, $\theta \in [0, 2\pi]$ equipped with the inner product defined as in (2.134):

$$\langle \zeta_1(r,\theta), \zeta_2(r,\theta) \rangle = \int_0^{2\pi} \int_0^{W_1} \zeta_1(r,\theta) \zeta_2^*(r,\theta) r \, dr \, d\theta, \tag{4.41}$$

where $*$ indicates the complex conjugate. Because of the missing orthogonality, modes with the same n have amplitudes $\Phi_{nm}(\omega)$ coupled together, meaning that their calculation cannot be separated as seen in section 2.8 for the liquid cylinder with rigid boundary. The first decoupling method proposed in this section is the *distance minimisation*. The underlying idea is to find a set of coefficients $\tilde{\Phi}_{nm}(\omega)$ that provide a good approximation of the pressure distribution by minimising an *objective function* conveniently defined. To define the objective function, a definition of *distance* is first introduced. Using the definition of inner product (4.41), the distance between two functions $\zeta_1(r,\theta)$ and $\zeta_2(r,\theta)$ can be defined as:

$$\Delta^2(\zeta_1,\zeta_2) = \frac{1}{\pi W_1^2}\big\langle(\zeta_1-\zeta_2),(\zeta_1-\zeta_2)\big\rangle \tag{4.42}$$
$$= \frac{1}{\pi W_1^2}\int_0^{2\pi}\int_0^{W_1}|\zeta_1(r,\theta)-\zeta_2(r,\theta)|^2 r\,dr\,d\theta,$$

which returns a non-negative scalar that measures the similarity between ζ_1 and ζ_2. Obviously, when ζ_1 and ζ_2 are equal, their distance is null. Assuming $\zeta_1 = \hat{p}_P(r,\theta,\omega)$ as the given source pressure distribution in $z=0$, and $\zeta_2 = \hat{\tilde{p}}_P(r,\theta,\omega)$ as its modal pressure approximation, the previous equation can be rewritten as:

$$\Delta^2(\hat{p}_P,\hat{\tilde{p}}_P) = \tag{4.43}$$
$$\frac{1}{\pi W_1^2}\int_0^{2\pi}\int_0^{W_1}\Big|\hat{p}_P(r,\theta,\omega)-\rho\omega^2\sum_{n=-\infty}^{+\infty}\sum_{m=1}^{+\infty}\tilde{\Phi}_{nm}J_n(q_{\phi_{nm}}r)e^{in\theta}\Big|^2 r\,dr\,d\theta,$$

where Δ^2 is a function of the frequency ω and of all the required amplitudes $\tilde{\Phi}_{nm}(\omega)$. Hence, per each frequency, ω, $\Delta^2(\tilde{\Phi}_{nm})$ is the required objective function, and ideally, the minimisation algorithm should find those values of $\tilde{\Phi}_{nm}$ for which Δ^2 is null. To be practically feasible, the infinite series should be truncated including only a finite number of coefficients $\tilde{\Phi}_{nm}$. In this case, the best possible outcome is finding those values of $\tilde{\Phi}_{nm}$ for which the function Δ^2 reaches its minimum value.

The main issue in implementing the minimisation algorithm is avoiding dummy solutions given by local minima. Therefore, it is convenient to reduce the number of coefficients involved by running the algorithm once per each value of n. Defining $\hat{\tilde{p}}_{\bar{n}}$ as:

$$\hat{\tilde{p}}_{\bar{n}}(r,\theta,\omega) = \rho\omega^2\sum_{m=1}^{+\infty}\tilde{\Phi}_{\bar{n}m}J_{\bar{n}}(q_{\phi_{\bar{n}m}}r)e^{i\bar{n}\theta}, \tag{4.44}$$

the distance between the source $\hat{p}_P$ and the approximation of the $\bar{n}^{th}$ pressure component $\hat{\tilde{p}}_{\bar{n}}$ is:

$$\Delta^2(\hat{p}_P,\hat{\tilde{p}}_{\bar{n}}) = \frac{1}{\pi W_1^2}\int_0^{2\pi}\int_0^{W_1}(\hat{p}_P-\hat{\tilde{p}}_{\bar{n}})(\hat{p}_P^*-\hat{\tilde{p}}_{\bar{n}}^*)r\,dr\,d\theta. \tag{4.45}$$

Using the source expansion (4.39) and the expression for $\hat{\tilde{p}}_{\bar{n}}$ (4.44) yields:

$$\Delta^2(\hat{p}_P, \hat{\tilde{p}}_{\bar{n}}) = \tag{4.46}$$
$$\frac{\rho^2\omega^4}{\pi W_1^2}\int_0^{2\pi}\int_0^{W_1}\Big[\sum_{k=-\infty}^{+\infty}\sum_{j=1}^{+\infty}\Phi_{kj}J_k(q_{\phi_{kj}}r)e^{ik\theta} - \sum_{m=1}^{+\infty}\tilde{\Phi}_{\bar{n}m}J_{\bar{n}}(q_{\phi_{\bar{n}m}}r)e^{i\bar{n}\theta}\Big]$$
$$\cdot\Big[\sum_{h=-\infty}^{+\infty}\sum_{u=1}^{+\infty}\Phi^*_{hu}J_h(q_{\phi_{hu}}r)e^{-ih\theta} - \sum_{l=1}^{+\infty}\tilde{\Phi}^*_{\bar{n}l}J_{\bar{n}}(q_{\phi_{\bar{n}l}}r)e^{-i\bar{n}\theta}\Big]r\,dr\,d\theta.$$

It is remarked here that, in the previous equation, Φ_{kj} and Φ_{hu} are the coefficients of the exact pressure expansion, while $\tilde{\Phi}_{\bar{n}m}$ and $\tilde{\Phi}_{\bar{n}l}$ are the coefficients of the best approximation sought for the index $\bar{n}$. Different indexes have been used for clarity. Considering that the integral in $d\theta$ is null over multiple of 2π periods, and separating the summation in k and h, the previous equation yields:

$$\Delta^2(\hat{p}_P, \hat{\tilde{p}}_{\bar{n}}) = \tag{4.47}$$
$$\frac{2\rho^2\omega^4}{W_1^2}\int_0^{W_1}\sum_{k=-\infty}^{\bar{n}-1}\Big[\sum_{j=1}^{+\infty}\Phi_{kj}J_k(q_{\phi_{kj}}r)\Big]\Big[\sum_{u=1}^{+\infty}\Phi^*_{ku}J_k(q_{\phi_{ku}}r)\Big]r\,dr$$
$$+\frac{2\rho^2\omega^4}{W_1^2}\int_0^{W_1}\sum_{k=\bar{n}+1}^{+\infty}\Big[\sum_{j=1}^{+\infty}\Phi_{kj}J_k(q_{\phi_{kj}}r)\Big]\Big[\sum_{u=1}^{+\infty}\Phi^*_{ku}J_k(q_{\phi_{ku}}r)\Big]r\,dr$$
$$+\frac{\rho^2\omega^4}{\pi W_1^2}\int_0^{2\pi}\int_0^{W_1}\Big[\sum_{j=1}^{+\infty}\Phi_{\bar{n}j}J_{\bar{n}}(q_{\phi_{\bar{n}j}}r)e^{i\bar{n}\theta} - \sum_{m=1}^{+\infty}\tilde{\Phi}_{\bar{n}m}J_{\bar{n}}(q_{\phi_{\bar{n}m}}r)e^{i\bar{n}\theta}\Big]$$
$$\cdot\Big[\sum_{u=1}^{+\infty}\Phi^*_{\bar{n}u}J_{\bar{n}}(q_{\phi_{\bar{n}u}}r)e^{-i\bar{n}\theta} - \sum_{l=1}^{+\infty}\tilde{\Phi}^*_{\bar{n}l}J_{\bar{n}}(q_{\phi_{\bar{n}l}}r)e^{-i\bar{n}\theta}\Big]r\,dr\,d\theta,$$

where the term:

$$\Delta^2(\hat{p}_{\bar{n}}, \hat{\tilde{p}}_{\bar{n}}) = \tag{4.48}$$
$$\frac{\rho^2\omega^4}{\pi W_1^2}\int_0^{2\pi}\int_0^{W_1}\Big[\sum_{j=1}^{+\infty}\Phi_{\bar{n}j}J_{\bar{n}}(q_{\phi_{\bar{n}j}}r)e^{i\bar{n}\theta} - \sum_{m=1}^{+\infty}\tilde{\Phi}_{\bar{n}m}J_{\bar{n}}(q_{\phi_{\bar{n}m}}r)e^{i\bar{n}\theta}\Big]$$
$$\cdot\Big[\sum_{u=1}^{+\infty}\Phi^*_{\bar{n}u}J_{\bar{n}}(q_{\phi_{\bar{n}u}}r)e^{-i\bar{n}\theta} - \sum_{l=1}^{+\infty}\tilde{\Phi}^*_{\bar{n}l}J_{\bar{n}}(q_{\phi_{\bar{n}l}}r)e^{-i\bar{n}\theta}\Big]r\,dr\,d\theta$$

is the distance between $\hat{p}_{\bar{n}}$ and $\hat{\tilde{p}}_{\bar{n}}$. Hence, since $\Delta(\hat{p}_{\bar{n}}, \hat{\tilde{p}}_{\bar{n}}) \geq 0$ and it is null only when $\tilde{\Phi}_{\bar{n}m} = \Phi_{\bar{n}m}$, it follows that the function $\Delta(\hat{p}_P, \hat{\tilde{p}}_{\bar{n}})$ reaches its minimum for those values of $\tilde{\Phi}_{\bar{n}m}$ that best approximate $\hat{p}_{\bar{n}}$. Therefore, the minimisation algorithm can be applied independently per each value of n, reducing the number of variables that need to be minimised simultaneously. This approach reduces the processing time and increases the accuracy.

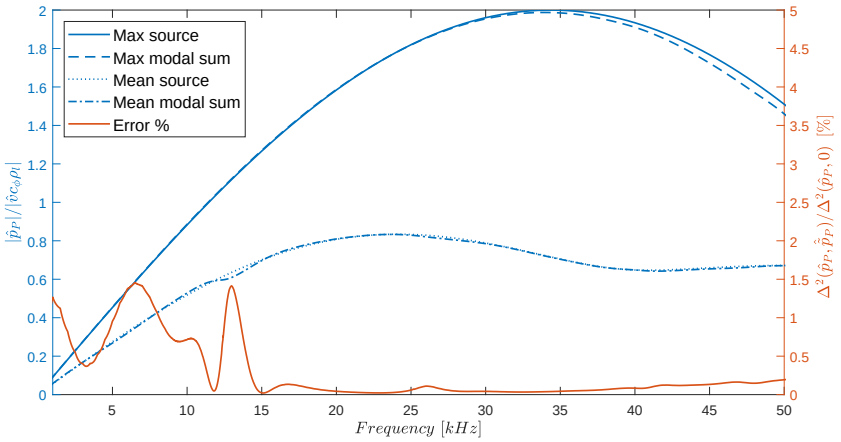


Figure 4.24: *Scaled pressure emitted from a circular piston of radius $R_P = 0.0225m$ in axisymmetric position $C_P \equiv (r_c = 0m, \theta_c = 0^\circ)$. In blue, dimensionless maximum and mean pressure values of the pressure source and of the related modal approximations obtained by distance minimisation on 14 modes. The red line reports the error as the normalised distance between the source and its modal approximation.*

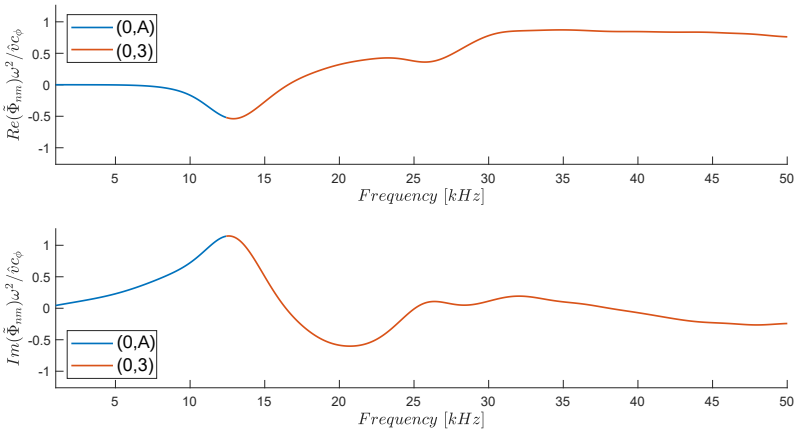


Figure 4.25: *Real and imaginary parts of the dimensionless scaled amplitudes for propagative mode $(0,3)$ and evanescent mode $(0,A)$. Values obtained by distance minimisation for a circular piston of radius $R_P = 0.0225m$ in axisymmetric position $C_P \equiv (r_c = 0.02m, \theta_c = 0^\circ)$.*

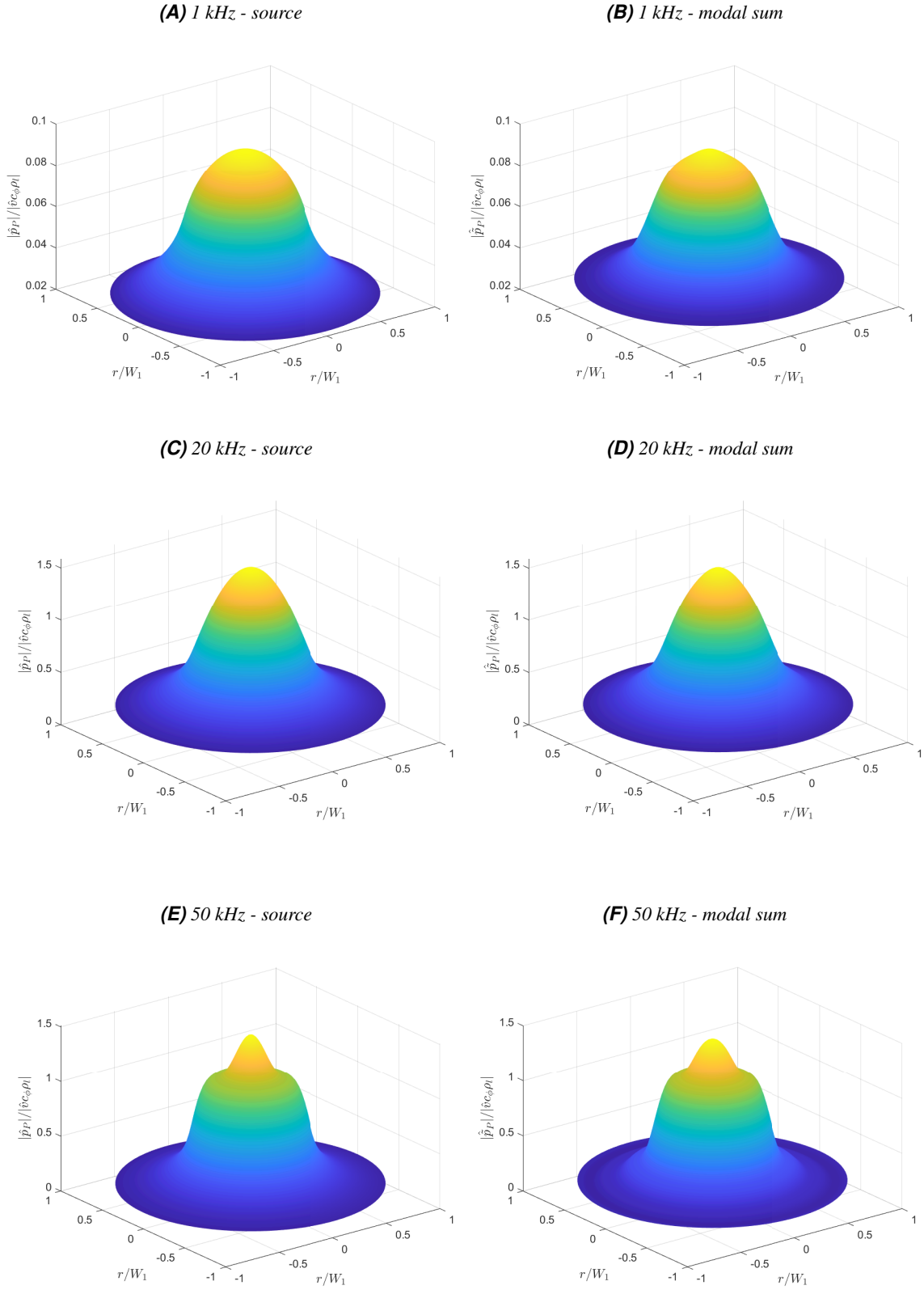


Figure 4.26: *Modal approximation of a radiating circular piston of radius $R_P = 0.0225m$ in axisymmetric position $C_P \equiv (r_c = 0m, \theta_c = 0°)$ obtained by distance minimisation on 14 modes. On the left-hand side, the dimensionless scaled source pressure distribution $\hat{p}_P$. On the right-hand side, the related approximated modal summations $\hat{\hat{p}}_P$.*

It is important to note that modes included in the objective function should be both propagative and evanescent. Indeed, the pressure approximation is performed in $z = 0$, where evanescent modes offer a contribution that is not negligible to the total pressure approximation.

Figure 4.24 reports the scaled dimensionless pressure amplitudes of a piston placed in axisymmetric position. Maximum and average pressures are divided by the scale factor $\hat{v}c_\phi\rho_l$, where $\hat{v}(\omega)$ is the piston velocity along z, c_ϕ the speed of sound in water and ρ_l the density of water. Overall, the maximum and the mean values of the modal approximation are very close to the desired values. The maximum value exhibits a slightly higher inaccuracy just in the last part of the frequency range considered. The red line reports the global relative error calculated as the distance between the pressure source distribution and its modal approximation. Clearly, the error is low everywhere, with higher values at very low frequencies and towards the high-frequency range. Figure 4.25 reports the real and imaginary parts of a couple of related scaled amplitudes. As shown, the scaled evanescent mode $(0,A)$ exists up to the cut-on frequency of the scaled propagative mode $(0,3)$. At the cut-off/cut-on, the scaled amplitudes of propagative and evanescent modes are the same. Other modes exhibit similar behaviour and are not reported for simplicity and brevity. Figures 4.26 provide a direct visual comparison between the scaled pressure source $\hat{p}_P$ (figures on the left-hand side) and the related scaled modal approximations $\hat{\hat{p}}_P$ (figures on the right-hand side). For all the frequencies reported, differences are barely detectable.

4.3.4 Integration of the source by analytical decoupling

As seen in the previous section, the integration of the source into the acoustic model is complicated by the interdependence of the modes. The distance minimisation method aims to overcome this issue by determining sets of coefficients that approximate the pressure distribution in $z = 0$. Nevertheless, this approach can be time-consuming in terms of calculation complexity and a second different approach is proposed in this section. Recalling equation (4.39), the pressure $\hat{p}_P$ in $z = 0$ can be approximated using a limited number of modes as:

$$\hat{p}_P(r,\theta,\omega) \approx \sum_{n=-N}^{+N} \hat{p}_n(r,\theta,\omega) = \rho\,\omega^2 \sum_{n=-N}^{+N}\sum_{m=1}^{+M} \Phi_{nm} J_n\left(q_{\phi_{nm}} r\right) e^{in\theta} = \hat{\tilde{p}}_P(r,\theta,\omega), \tag{4.49}$$

where n and m are integers in the range $[-N:N]$ and $[1:M]$, respectively. Multiplying both members of the previous equation by the *decoupling factor*:

$$J_h\left(q_{\phi_{hj}} r\right) e^{ih\theta} \tag{4.50}$$

and integrating over the section of the waveguide, with $r \in [0, W_1]$ and $\theta \in [0, 2\pi]$, yields:

$$\int_0^{2\pi}\int_0^{W_1} \hat{p}_P(r,\theta,\omega) J_h\left(q_{\phi_{hj}} r\right) e^{ih\theta} r\,dr\,d\theta \approx \rho\,\omega^2 \sum_{n=-N}^{+N}\sum_{m=1}^{+M} \Phi_{nm} \int_0^{2\pi}\int_0^{W_1} J_n\left(q_{\phi_{nm}} r\right) J_h\left(q_{\phi_{hj}} r\right) e^{i(n+h)\theta} r\,dr\,d\theta, \tag{4.51}$$

where indexes h and j are chosen in the range $[-N:N]$ and $[1:M]$, respectively. The right-hand side of the previous equation is not null only for $h = -n$ since the integral in $d\theta$ is null over an integer number of periods. Assuming $h = -n$ and noting that $q_{nm}(\omega) = q_{-nm}(\omega)$, the previous equation can be rewritten as:

$$\int_0^{2\pi}\int_0^{W_1} \hat{p}_P(r,\theta,\omega) J_{-n}\left(q_{\phi_{nj}} r\right) e^{-in\theta} r\,dr\,d\theta \approx 2\pi\rho\,\omega^2 \sum_{m=1}^{+M} \Phi_{nm} \int_0^{W_1} J_n\left(q_{\phi_{nm}} r\right) J_{-n}\left(q_{\phi_{nj}} r\right) r\,dr. \tag{4.52}$$

For any ω and per each n, the previous equation contains M unknowns $\Phi_{nm}(\omega)$ coupled together. Besides, M different equations can be obtained by choosing the index j in the range $[1:M]$. Hence, for any ω and per each $n \in [-N:N]$, (4.52) represents a system of M equations in M unknowns. Note that only one value of n appears in each system, so the equations are decoupled in the θ direction. From the above, the decoupling systems of equations can be written as:

$$\mathbf{\Lambda}(n,\omega)\mathbf{\Phi}(n,\omega) \approx \frac{1}{2\pi\rho\omega^2}\mathbf{\Upsilon}(n,\omega), \tag{4.53}$$

which can be expanded as:

$$\begin{bmatrix} \int_0^{W_1} J_n(q_{\phi n1}r)J_{-n}(q_{\phi n1}r)rdr & \dots & \int_0^{W_1} J_n(q_{\phi nm}r)J_{-n}(q_{\phi n1}r)rdr & \dots & \int_0^{W_1} J_n(q_{\phi nM}r)J_{-n}(q_{\phi n1}r)rdr \\ \dots & \dots & \dots & \dots & \dots \\ \int_0^{W_1} J_n(q_{\phi n1}r)J_{-n}(q_{\phi nj}r)rdr & \dots & \int_0^{W_1} J_n(q_{\phi nm}r)J_{-n}(q_{\phi nj}r)rdr & \dots & \int_0^{W_1} J_n(q_{\phi nM}r)J_{-n}(q_{\phi nj}r)rdr \\ \dots & \dots & \dots & \dots & \dots \\ \int_0^{W_1} J_n(q_{\phi n1}r)J_{-n}(q_{\phi nM}r)rdr & \dots & \int_0^{W_1} J_n(q_{\phi nm}r)J_{-n}(q_{\phi nM}r)rdr & \dots & \int_0^{W_1} J_n(q_{\phi nM}r)J_{-n}(q_{\phi nM}r)rdr \end{bmatrix}$$

$$\cdot \begin{bmatrix} \Phi_{n1} \\ \dots \\ \Phi_{nm} \\ \dots \\ \Phi_{nM} \end{bmatrix} \approx \frac{1}{2\pi\rho\omega^2} \begin{bmatrix} \int_0^{2\pi}\int_0^{W_1} p_P J_{-n}(q_{\phi n1}r)e^{-in\theta} r\,dr\,d\theta \\ \dots \\ \int_0^{2\pi}\int_0^{W_1} p_P J_{-n}(q_{\phi nj}r)e^{-in\theta} r\,dr\,d\theta \\ \dots \\ \int_0^{2\pi}\int_0^{W_1} p_P J_{-n}(q_{\phi nM}r)e^{-in\theta} r\,dr\,d\theta \end{bmatrix} \forall n \in [0:N]. \tag{4.54}$$

To calculate the vector of coefficients $\mathbf{\Phi}(n,\omega)$, the squared matrix $\mathbf{\Lambda}(n,\omega)$ should be calculated and then inverted. This operation is repeated per each n and for any ω in the desired range. As for the distance minimisation, for axisymmetric sources, the decomposition requires only $n = 0$ modes.

4.3.4.1 Calculation of the modal amplitudes

Finding $\mathbf{\Phi}(n,\omega)$ by simple inversion of the matrix $\mathbf{\Lambda}(n,\omega)$, as mentioned in the previous section, hides an important issue. Indeed, for several frequencies, $\mathbf{\Lambda}(n,\omega)$ is almost singular and tiny variations of $\mathbf{\Upsilon}(n,\omega)$ can generate huge variations in $\mathbf{\Phi}(n,\omega)$. The impact is so severe that even the numerical rounding (due to the finite number of digits in a standard calculator) can generate results mainly dominated by errors. Problems with this kind of issue are called *ill-conditioned* [240, p. 126], and several methods to improve the accuracy of the solution can be found in the literature [241],[242]. One of the most widely used is the *truncated singular value decomposition (TSVD)*. The main idea is to generate a new matrix $\tilde{\mathbf{\Lambda}}(n,\omega)$

that provides a solution close to the one sought and robust with respect to the inaccuracies introduced by the error in $\boldsymbol{\Upsilon}(n,\omega)$. This procedure is generally called *regularisation* [243]. Using the *singular value decomposition (SVD)* and temporarily omitting n and ω, $\boldsymbol{\Lambda}$ can be decomposed as follow:

$$\boldsymbol{\Lambda} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T = [\boldsymbol{u_1}\boldsymbol{u_2}...\boldsymbol{u_M}] \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_M \end{bmatrix} [\boldsymbol{v_1}\boldsymbol{v_2}...\boldsymbol{v_M}]^T, \tag{4.55}$$

where $\boldsymbol{\Sigma}$ is the diagonal matrix of the *singular values*, and $\boldsymbol{U}$ and $\boldsymbol{V}$ are the orthonormal left and right *singular vectors*, respectively. Vectors $\boldsymbol{u_m}$ are the eigenvectors of $\boldsymbol{\Lambda}\boldsymbol{\Lambda}^T$, vectors $\boldsymbol{v_m}$ are the eigenvectors of $\boldsymbol{\Lambda}^T\boldsymbol{\Lambda}$, and the singular values σ_m are the square roots of the eigenvalues of $\boldsymbol{\Lambda}\boldsymbol{\Lambda}^T$ or $\boldsymbol{\Lambda}^T\boldsymbol{\Lambda}$. When building the matrices, the singular values σ_m are sorted in descending order:

$$\sigma_1 \geq \sigma_1 \geq ... \geq \sigma_M \geq 0. \tag{4.56}$$

Plugging (4.55) into (4.53), the vector of modal amplitudes $\boldsymbol{\Phi}(n,\omega)$ can be written as:

$$\boldsymbol{\Phi} \approx \frac{1}{2\pi\rho\omega^2}\boldsymbol{U}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{V}\boldsymbol{\Upsilon}, \tag{4.57}$$

where $\boldsymbol{\Sigma}^{-1}$ is built by using the inverse of each singular value $1/\sigma_m$. This form provides a further explanation. In determining $\boldsymbol{\Phi(n,\omega)}$, a very small singular value introduces a correspondent very high value in $\boldsymbol{\Sigma}^{-1}$, which, in turn, causes a great sensibility of the solution to the error in $\boldsymbol{\Upsilon}(n,\omega)$. On the contrary, assuming the exact solution $\boldsymbol{\Phi}(n,\omega)$ is known and using (4.53) and (4.55), smaller singular values may offer a negligible contribution to $\boldsymbol{\Upsilon}(n,\omega)$. Hence, the TSVD aims to calculate an approximation of the modal amplitudes, namely $\tilde{\boldsymbol{\Phi}}(n,\omega)$, by neglecting the contribution of the smallest singular values. Note that the notation $\tilde{\boldsymbol{\Phi}}(n,\omega)$ also indicates the approximation related to the truncation of the modal series to M modes. To achieve that, a new matrix $\tilde{\boldsymbol{\Sigma}}_v(n,\omega)$ is obtained from $\boldsymbol{\Sigma}(n,\omega)$ by replacing the last $M-v$ terms along the diagonal with zeros:

$$\tilde{\boldsymbol{\Sigma}}_\nu = \begin{bmatrix} \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_\nu & \\ & & & 0 \end{bmatrix}. \tag{4.58}$$

Replacing $\boldsymbol{\Sigma}(n,\omega)$ with $\tilde{\boldsymbol{\Sigma}}_\nu(n,\omega)$ in (4.55) and plugging it into (4.53) yields:

$$\tilde{\boldsymbol{\Lambda}}_\nu \tilde{\boldsymbol{\Phi}} = \frac{1}{2\pi\rho\omega^2}\boldsymbol{\Upsilon}. \tag{4.59}$$

From the previous equation, the approximated modal amplitudes $\tilde{\boldsymbol{\Phi}}$ can be calculated as:

$$\tilde{\boldsymbol{\Phi}} = \frac{1}{2\pi\rho\omega^2}\tilde{\boldsymbol{\Lambda}}_\nu^\dagger \boldsymbol{\Upsilon}, \tag{4.60}$$

where the symbol † indicates the *Moore-Penrose pseudoinverse* [244]. Note that $\tilde{\boldsymbol{\Lambda}}_\nu$ cannot be simply inverted since its rank is ν. Besides, $\tilde{\boldsymbol{\Lambda}}_\nu^\dagger$ can be written as:

$$\tilde{\boldsymbol{\Lambda}}_\nu^\dagger = \boldsymbol{U}^T \tilde{\boldsymbol{\Sigma}}_\nu^\dagger \boldsymbol{V}, \tag{4.61}$$

where:

$$\tilde{\boldsymbol{\Sigma}}_\nu^\dagger = \begin{bmatrix} 1/\sigma_1 & & & \\ & \ddots & & \\ & & 1/\sigma_\nu & \\ & & & 0 \end{bmatrix}. \tag{4.62}$$

Therefore, the approximation of the modal amplitudes $\tilde{\boldsymbol{\Phi}}$ is obtained by removing those factors that make the solution sensible to the approximations in $\boldsymbol{\Upsilon}$. A key issue of the TSVD is the determination of the truncation index ν. Although several optimisation techniques are reported in the literature [243],[245], the optimal value for ν is here calculated using a simplified approach aimed to keep the calculation complexity low. The *condition number* is defined as:

$$\kappa(\mathbf{\Lambda}) = \frac{\sigma_1}{\sigma_M}, \tag{4.63}$$

and, from its definition, the higher the condition number, the more sensible $\tilde{\mathbf{\Phi}}$ is to the approximations in $\mathbf{\Upsilon}$ [243]. Using the condition number, it is possible to give a measure of the conditioning problem and, at the same time, to assess the truncation index ν. Assuming $\delta\tilde{\mathbf{\Phi}}$ as the error of the modal amplitude related to the approximations $\delta\mathbf{\Upsilon}$, applying the *Euclidean vector norm*[5] and the related properties[6], the two following relations hold:

$$\left\|\mathbf{\Lambda}\right\|_2 \left\|\tilde{\mathbf{\Phi}}\right\|_2 \geq \frac{1}{2\pi\rho\omega^2} \left\|\mathbf{\Upsilon}\right\|_2 \tag{4.64}$$

$$\left\|\delta\tilde{\mathbf{\Phi}}\right\|_2 \leq \frac{1}{2\pi\rho\omega^2} \left\|\mathbf{\Lambda}^\dagger\right\|_2 \left\|\delta\mathbf{\Upsilon}\right\|_2, \tag{4.65}$$

where the norms applied to $\mathbf{\Lambda}(n,\omega)$ and $\mathbf{\Lambda}(n,\omega)^\dagger$ are the associated *induced matrix norms*.

Dividing (4.65) by $\left\|\tilde{\mathbf{\Phi}}\right\|_2$ and using (4.64) yields:

$$\frac{\left\|\delta\tilde{\mathbf{\Phi}}\right\|_2}{\left\|\tilde{\mathbf{\Phi}}\right\|_2} \leq \left\|\mathbf{\Lambda}\right\|_2 \left\|\mathbf{\Lambda}^\dagger\right\|_2 \frac{\left\|\delta\mathbf{\Upsilon}\right\|_2}{\left\|\mathbf{\Upsilon}\right\|_2}, \tag{4.66}$$

which, using definition (4.63), can be rewritten as:

$$\frac{\left\|\delta\tilde{\mathbf{\Phi}}\right\|_2}{\left\|[\tilde{\mathbf{\Phi}}\right\|_2} \leq \kappa(\mathbf{\Lambda}) \frac{\left\|\delta\mathbf{\Upsilon}\right\|_2}{\left\|\mathbf{\Upsilon}\right\|_2}. \tag{4.67}$$

Since the input error $\left\|\delta\mathbf{\Upsilon}\right\|_2 / \left\|\mathbf{\Upsilon}\right\|_2$ is fixed by the relative accuracy of the calculations, the error of the modal amplitude can be reduced by reducing the ratio of the maximum singular value to the minimum non-null singular value[7].

[5] The *Euclidean norm* for a vector $\boldsymbol{x} = [x_1, x_2, ..., x_M]$ is defined as $\left\|\boldsymbol{x}\right\|_2 = \sqrt{x_1^2 + x_2^2 + ... + x_M^2}$, while the *associated induced matrix norm* is simply equal to the largest singular value σ_1.

[6] Vector and matrix norms satisfy the properties: $\left\|\boldsymbol{AB}\right\|_2 \leq \left\|\boldsymbol{A}\right\|_2 \left\|\boldsymbol{B}\right\|_2$, $\left\|\alpha\boldsymbol{A}\right\|_2 = |\alpha| \left\|\boldsymbol{A}\right\|_2$

[7] Note that the condition number of $\tilde{\mathbf{\Lambda}}_k$ is infinite.

However, since [243]:

$$\left\|\boldsymbol{\Lambda}-\tilde{\boldsymbol{\Lambda}}_{\nu}\right\|_2=\sigma_{\nu+1}, \tag{4.68}$$

a truncation index too small improves the conditioning but offers an inaccurate approximation of $\boldsymbol{\Lambda}$. Therefore, the truncation index should be determined as a trade-off between the conditioning and the approximation of $\boldsymbol{\Lambda}$. For practical calculation purposes, $\left\|\delta\boldsymbol{\Upsilon}\right\|_2$ can be assigned as the norm of the *floating point relative accuracy*[8] of $\boldsymbol{\Upsilon}$. The truncation index ν is determined from:

$$\frac{\sigma_1}{\sigma_\nu}\leq w_f\frac{\left\|\boldsymbol{\Upsilon}\right\|_2}{\left\|eps(\boldsymbol{\Upsilon})\right\|_2}, \tag{4.69}$$

where the weight w_f is defined according to the desired final error of the approximated amplitudes $\tilde{\boldsymbol{\Phi}}$. Note that w_f affects the range of the small singular values retained, and consequently, it influences the spatial dynamics associated with small modal components. Moreover, the final error on $\tilde{\boldsymbol{\Phi}}$ depends also on the initial number of modes included in the approximation.

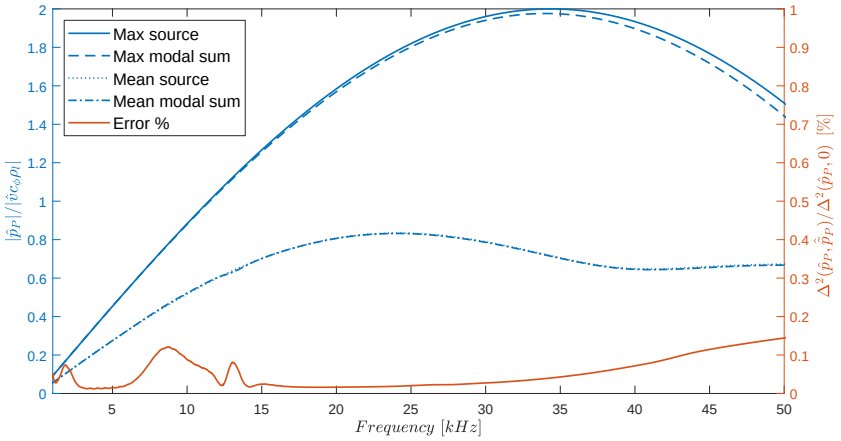
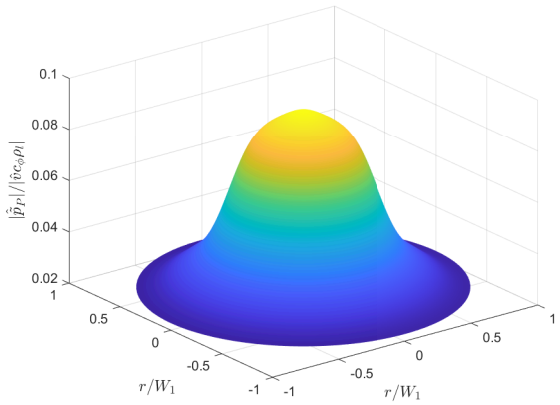


Figure 4.27: *Dimensionless scaled pressure radiated from a circular piston of radius $R_P=0.0225m$ in axisymmetric position $C_P\equiv(r_c=0m,\theta_c=0^\circ)$. In blue, the scaled dimensionless maximum and mean pressure values of the actual source and the related modal approximations obtained by analytical decoupling on 14 modes. In red, the error as the normalised distance between the source and its modal approximation.*

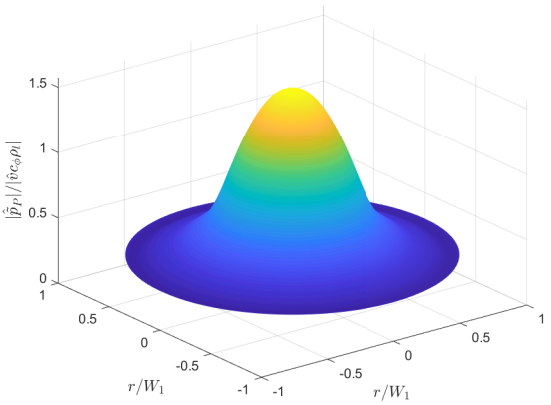
[8] Function *eps* in Matlab.

Figure 4.27 reports the accuracy results obtained by applying the analytical decoupling for the modal decomposition of the pressure emitted by a circular piston of radius $R_P = 0.0225m$ with centre $C_P \equiv (r_c, \theta_c)$ on the axis of the waveguide. Results in the range $0 - 50kHz$ can be directly compared with those reported in the previous section since the same number of modes for $n = 0$ have been used (7 propagative and 7 evanescent). Although the distance minimisation yields slightly better accuracy for the maximum pressure at higher frequencies, overall, both methods return max and mean values very close to those desired. The relative error for the distance, however, despite being small in both cases, appears to be roughly one order of magnitude lower for the analytical decoupling.

***(A)** 1 kHz - modal sum*



***(B)** 20 kHz - modal sum*



***(C)** 50 kHz - modal sum*

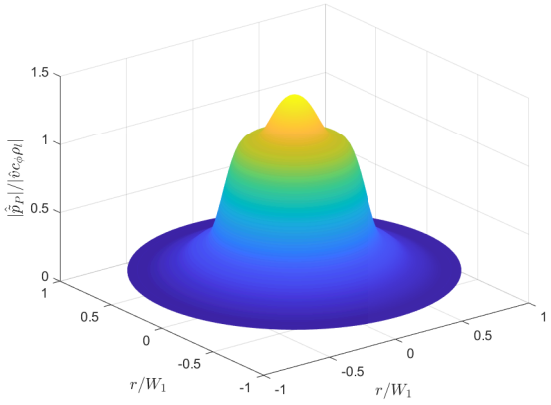


***Figure 4.28:** Modal approximations of a radiating axisymmetric circular piston of radius $R_P = 0.0225m$ placed in $C_P \equiv (r_c = 0m, \theta_c = 0°)$. Analytical decoupling obtained using 14 modes. Related source pressure distributions are reported in figures 4.26A, 4.26C, 4.26E.*

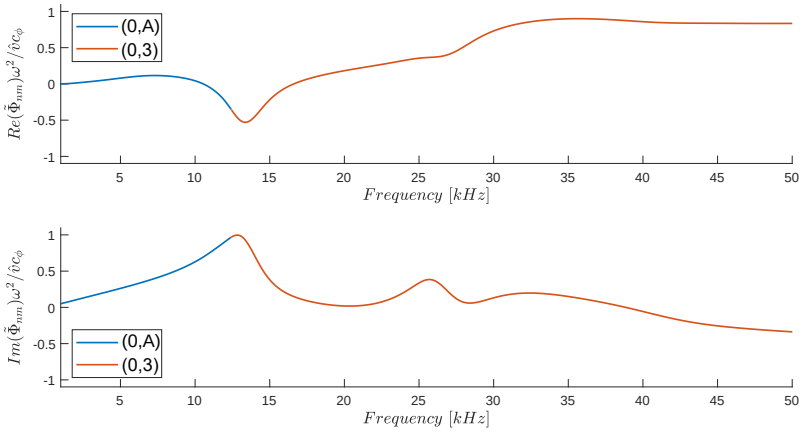


Figure 4.29: *Real and imaginary part of the scaled amplitudes for propagative mode* $(0,3)$ *and evanescent mode* $(0,A)$*. Values are obtained by analytical decoupling for a circular piston of radius* $R_P = 0.0225m$ *in axisymmetric position* $C_P \equiv (r_c = 0.02m, \theta_c = 0°)$*.*

As a result, the modal sum reported in figures 4.28 are barely distinguishable from the original pressure source distribution reported in 4.26A, 4.26C, 4.26E. Note that the amplitudes obtained by analytical decoupling, although not identical, are very close to those obtained by distance minimisation. An example for the modes $(0,A)$ and $(0,3)$ is reported in figure 4.29, which can be directly compared with the equivalent reported in figure 4.25.

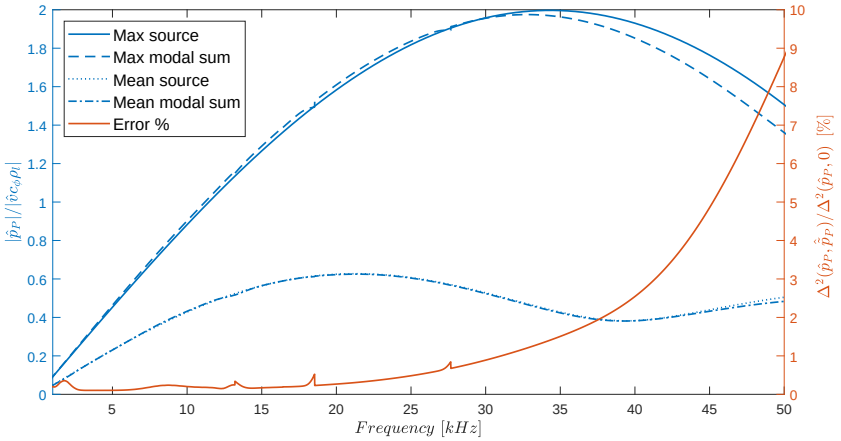


Figure 4.30: *Dimensionless scaled pressure radiated from a circular piston of radius* $R_P = 0.0225m$ *in non-axisymmetric position* $C_P \equiv (r_c = 0.02m, \theta_c = 0°)$*. In blue, the dimensionless maximum and mean pressure values of the actual source and the related modal approximations obtained by analytical decoupling on 110 modes. In red, the error as the normalised distance between the source and its modal approximation.*

Another example concerns the same circular piston as above but placed in a non-axisymmetric position with $r_c = 0.02m$ and $\theta_c = 0°$. Because of the missing symmetry, the modal approximation should also include $n \neq 0$ modes to account for the θ angular dependency. In addition to the 14 modes used in the previous case, further 96 modes ($14_{n=1}$, $14_{n=2}$, $14_{n=3}$ propagative and $18_{n=1}, 20_{n=2}, 16_{n=3}$ evanescent) are included for the decomposition. Indeed, a good approximation for a non-axisymmetric pressure distribution requires a much higher number of modes compared to the axisymmetric case. As reported in figure 4.30, despite the maximum and the mean values remaining close to the desired values, the distance error appears to be much higher, especially for higher frequencies (where the shape of the pressure distribution becomes more intricate).

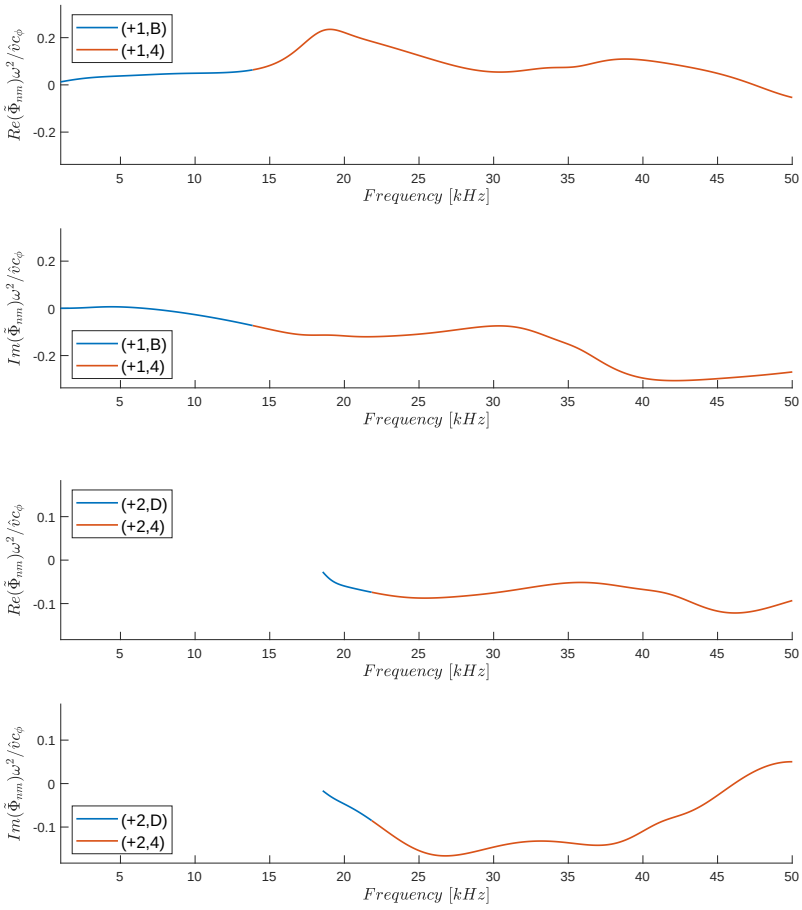


Figure 4.31: *Real and imaginary part of the scaled modal amplitudes obtained by analytical decoupling. Circular piston of radius $R_P = 0.0225m$ in non-axisymmetric position $C_P \equiv (r_c = 0.02m, \theta_c = 0°)$. Top - modes $(+1,4)/(+1,B)$. Bottom - modes $(+2,4)/(+2,D)$.*

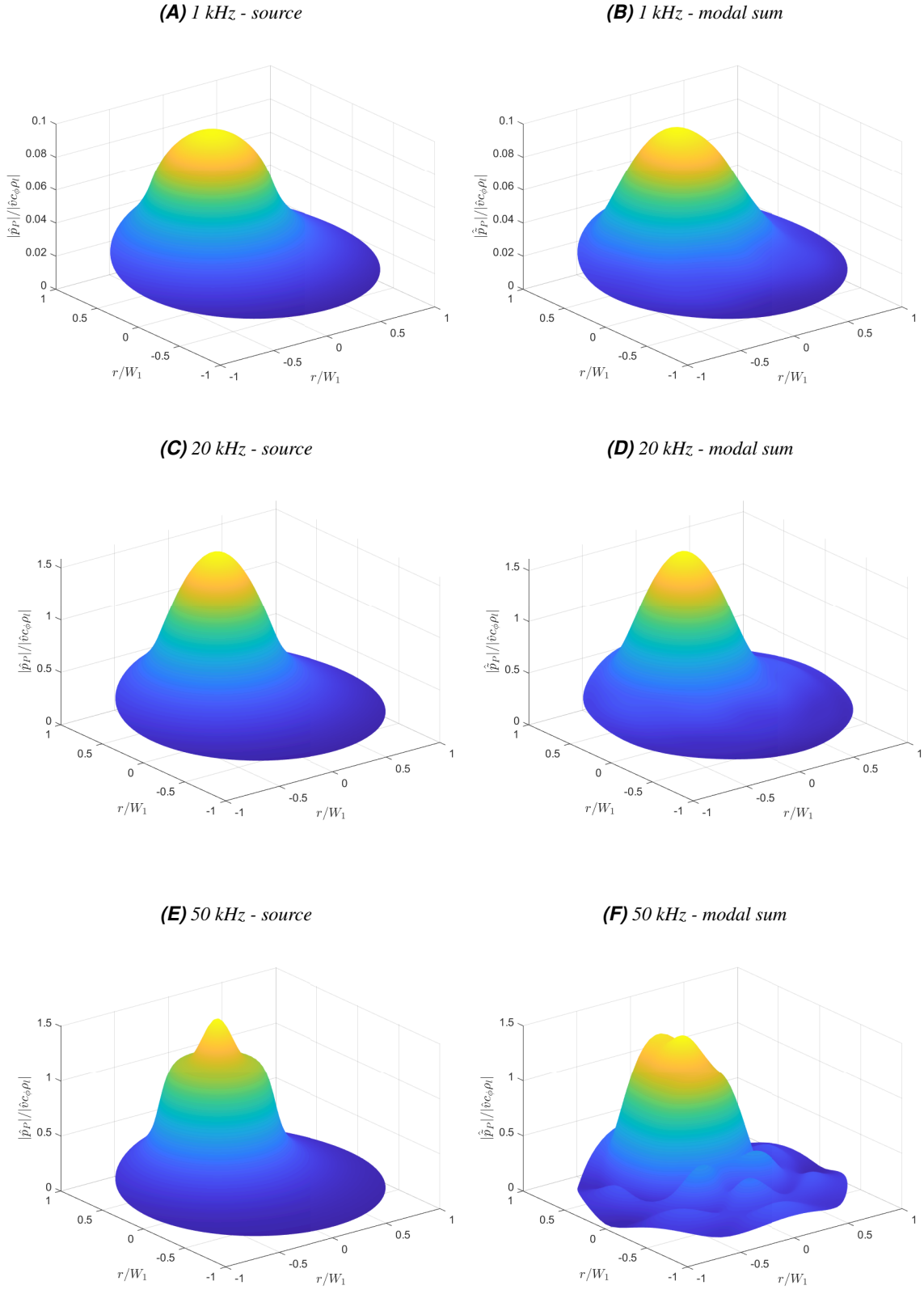


Figure 4.32: *Modal approximation of a radiating circular piston of radius $R_P = 0.0225m$ in non-axisymmetric position $C_P \equiv (r_c = 0.02m, \theta_c = 0°)$ obtained by analytical decoupling with 110 modes. On the left-hand side, the dimensionless scaled source pressure distribution. On the right-hand side, the related modal sum.*

A clear visualisation is provided in figures 4.32, where the modal sum at $50kHz$ can be easily distinguished from its original representation.

Two examples of modal amplitudes are reported in figure 4.31. As for distance minimisation, analytical decoupling returns continuous amplitude at those frequencies (cut-off/cut-on) where evanescent modes turn into propagative modes. However, as shown in figure 4.9 for $n = 2$ and figure 4.12 for $n = 3$, not all the evanescent modes have zero cut-on, and not all of them turn into propagative modes. Under these circumstances, the amplitudes of these evanescent modes tend to be null at the cut-on/cut-off transition. An example is reported in figure 4.31 for the scaled evanescent mode $(+2, D)$, whose amplitude is null at its evanescent cut-on.

4.3.5 Angular invariance

When the pressure source has a non-axisymmetric distribution, the value of the complex amplitude Φ_{nm} depends on the position of the source in the chosen reference system. Figure 4.33 illustrates the variation of the modal amplitudes of the example reported in figure 4.32, when the centre of the source $C_P \equiv (r_c, \theta_c)$ is rotated around the axis of the waveguide while keeping the receiver at the same position. It is shown how both real and imaginary parts of Φ_{nm} are described by the same sine functions with constant amplitudes and shift equal to a quarter of the period[9]. Indeed, for $\theta_c \in [0 \rightarrow 360^\circ]$, the modes can be written as:

$$\Phi_{nm}(\theta_c) = |\Phi_{nm}| e^{-in(\theta_c + \alpha_0)}, \tag{4.70}$$

with α_0 constant. Although the modulus is angular invariant, the phase changes linearly with period $2\pi/n$. However, since each mode exhibits an angular variation of the form $e^{in\theta}$ (with the position of the receiver), if the relative angular position between the source S_P and the receiver G is kept constant, their absolute angular position on the reference system has no influence in the determination of the modal pressure.

[9] Since a limited number of modes is accounted for the calculation, the shape of the variation calculated is only roughly sinusoidal.

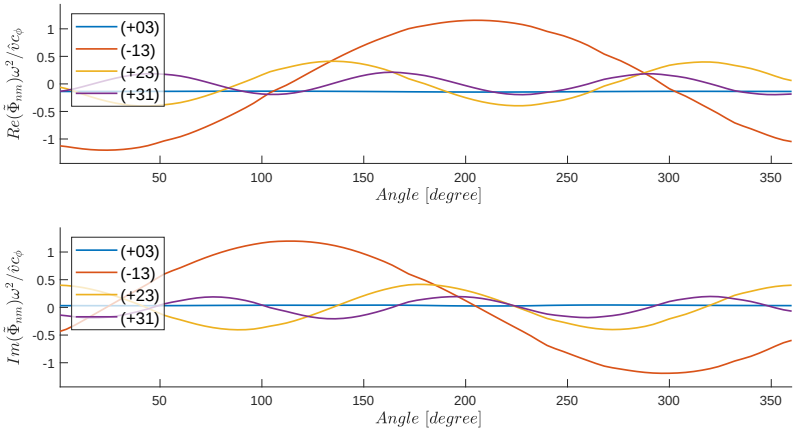


Figure 4.33: *Scaled modal amplitudes as a function of the angular position of the source ($n = 0, -1, +2, +3$). Circular piston of radius $R_P = 0.0225m$ at $25kHz$ in non-axisymmetric position. $r_c = 0.02m$ and $\theta_c \in [0 \rightarrow 360°]$.*

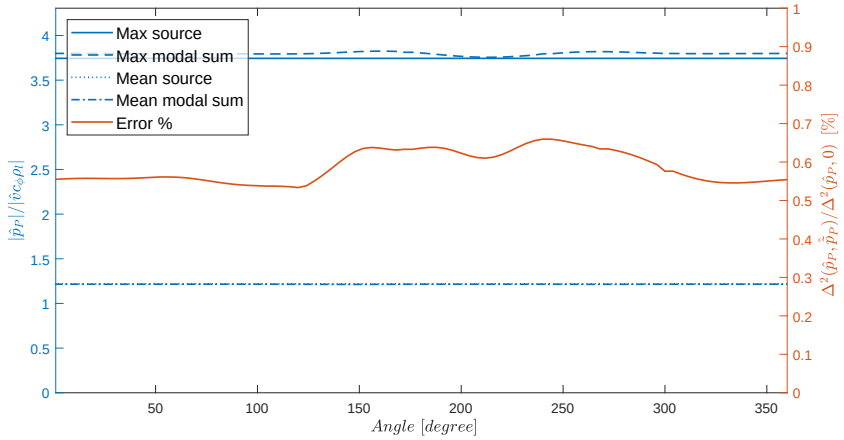


Figure 4.34: *Dimensionless pressure for a circular piston of radius $R_P = 0.0225m$ at $25kHz$ in non-axisymmetric position with $r_c = 0.02m$ and $\theta_c \in [0 \rightarrow 360°]$. In blue, the maximum and mean pressure values and the related modal approximations obtained by analytical decoupling on 110 modes. The red line reports the error as normalised distance between the source and its modal approximation.*

Therefore, the modal representation can be assumed *angular invariant* with respect to the reference system, a conclusion that is in accordance with the symmetry of the system. Figure 4.34 shows the maximum and the mean value of the source pressure distribution and the related modal approximations when the centre of the source $C_P \equiv (r_c, \theta_c)$ is shifted in the θ direction in respect of the

reference system. The red line reports the error as the relative distance between the source and its modal approximation. As can be observed, the approximation accuracy is pretty much constant for each value of θ_c, being the difference caused by the truncation of the modal series.

4.4 Modelling the output signal

The third layer of the model proposed in this chapter concerns the calculation of the reverberated signals in the time or the frequency domain. The output is evaluated at a point G in the inner liquid along the waveguide and at a certain distance from the source S_P placed in $z = 0$ (figure 4.1). A driving time or frequency domain input signal must be defined along with the quantities required by the first two layers of the model: geometry, materials, and pressure source distribution. The spectrum of the driving signal defines which modes are excited in a propagative form and which modes exist only in an evanescent form. The amplitudes of the modes depend on both the driving signal and the pressure distribution of the source. As seen in section 4.3, the evanescent modes need to be included in the second layer of the model for the calculation of the modal amplitudes. Indeed, their contribution is not negligible at the source. For the third layer, however, the distance from the source is assumed large enough to account for the exponential decay, and the evanescent modes can be neglected.

From sections 2.9 and 4.3, the approximated frequency domain representation of the pressure in the waveguide can be written as:

$$\hat{\tilde{p}}(r,\theta,z,\omega) = \rho\omega^2 \sum_{n=-N}^{+N} \sum_{m=1}^{+M_n} \tilde{\Phi}_{nm} J_n(q_{\phi_{nm}} r) e^{in\theta} e^{ik_{z_{nm}} z}. \tag{4.71}$$

In the previous equation, the approximated modal amplitudes $\tilde{\Phi}_{nm}(\omega)$ are only known in their scaled form $\bar{\Phi}_{nm} = \tilde{\Phi}_{nm}/\hat{v} c_\phi \rho$. Hence, dividing the (4.71) by the velocity $\hat{v}(\omega)$, it is possible to obtain the *transfer function* between output pressure and the driving velocity of the piston:

$$\hat{\tilde{p}}_v = \frac{\hat{\tilde{p}}(r,\theta,z,\omega)}{\hat{v}(\omega)} = \rho^2 c_\phi \omega^2 \sum_{n=-N}^{+N} \sum_{m=1}^{+M_n} \bar{\Phi}_{nm} J_n(q_{\phi_{nm}} r) e^{in\theta} e^{ik_{z_{nm}} z}. \tag{4.72}$$

Equation 4.72, although important, is not directly useful since the velocity of the piston is unknown. In general, it is necessary to establish a relationship between a known input and the output pressure. Therefore, named $\hat{x}(\omega)$ the desired input signal, a further transfer function $S(\omega) = \hat{v}(\omega)/\hat{x}(\omega)$ must be defined to obtain the desired transfer function:

$$\hat{\tilde{p}}_x(\omega) = S(\omega)\hat{\tilde{p}}_v = \frac{\hat{v}(\omega)}{\hat{x}(\omega)} \frac{\hat{\tilde{p}}(r,\theta,z,\omega)}{\hat{v}(\omega)} \tag{4.73}$$

The possible solutions for the definition of the transfer function $S(\omega)$ are analysed in section 5.5.1. For now, it can just be observed that, once the input signal $\hat{x}(\omega)$ has been defined, the output pressure at a point $G \equiv (r_G, \theta_G, z_G)$ can be calculated as:

$$\begin{aligned} \hat{\tilde{p}}(r_G,\theta_G,z_G,\omega) = \hat{x}(\omega)\hat{\tilde{p}}_x(r_G,\theta_G,z_G,\omega) = \\ \hat{x}(\omega)S(\omega)\rho^2 c_\phi \omega^2 \sum_{n=-N}^{+N} \sum_{m=1}^{+M_n} \bar{\Phi}_{nm} J_n(q_{\phi_{nm}} r_G) e^{in\theta_G} e^{ik_{z_{nm}} z_G} \end{aligned} \tag{4.74}$$

The related time-domain representation can be obtained by calculating the inverse Fourier transform of equation 4.74. Note that, since the inverse Fourier transform is a linear operator, the total pressure signal in the time domain is also given by the superimposition of each mode in the time domain.

As an example of signal processing, assuming a sample rate of $100kHz$, results of the first two layers of the model obtained in the frequency band between $0-50kHz$ and in steps of $10Hz$ can be used directly to process a signal $100ms$ long. If the input signal needs to be truncated, a window (which allows sufficient damping at the end of the recording) can be used to avoid spectral *leakage* [246]. When a simulation for a longer recording is needed, the input signal can be processed in short windows, or an interpolation of the results of the first two layers can be calculated. Further details are reported in section 5.5.

To visualise the results of the simulation, the dimensionless output pressure in G is defined as:

$$\hat{\tilde{p}}_0(r_G,\theta_G,z_G,\omega) = \frac{\hat{\tilde{p}}_x(r_G,\theta_G,z_G,\omega)}{S(\omega)c_\phi\rho}\hat{x}(\omega) = \hat{x}(\omega)\rho\omega^2 \sum_{n=-N}^{+N}\sum_{m=1}^{+M_n}\bar{\Phi}_{nm}J_n(q_{\phi_{nm}}r_G)e^{in\theta_G}e^{ik_{z_{nm}}z_G}, \tag{4.75}$$

where $\hat{x}(\omega)$ is assumed to be a dimensionless input signal. The following examples report $\tilde{p}_0$ for the pipe of figure 4.1 described in section 4.1.1.3 and for a piston of radius $R_P = 0.0225m$ in different configurations. In the first two cases, the source and the receiver are in non-axisymmetric positions, meaning that modes for $|n| > 0$ should be included. The dimensionless driving signal $x(t)$ is assumed to be the *chirp* signal represented in figure 4.35 with a frequency range spanning from $7.5kHz$ to $25kHz$.

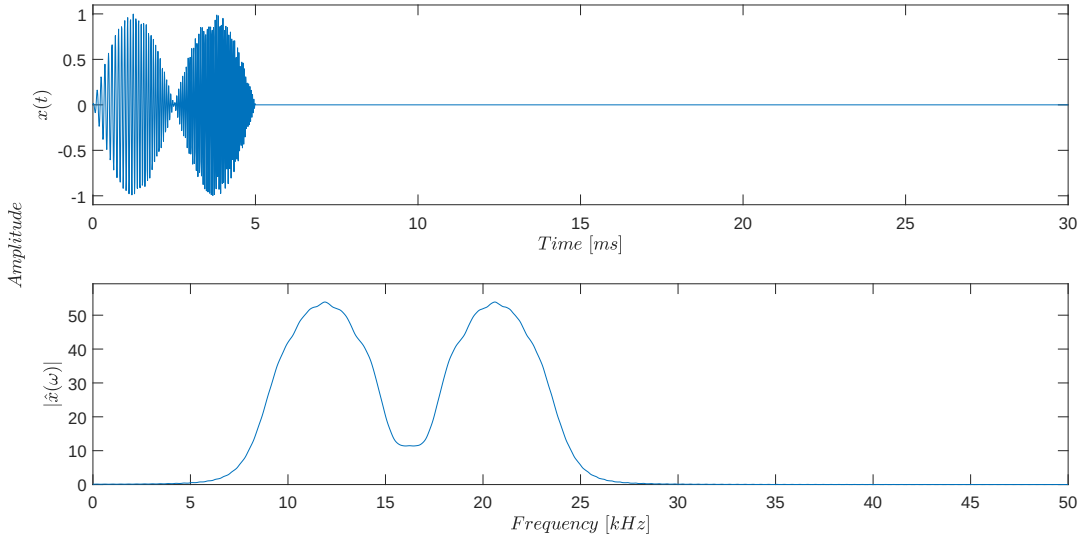


Figure 4.35: *Dimensionless chirp signal driving the pressure source. Time domain representation (top) and frequency spectrum (bottom).*

Figure 4.36 reports the output signal when the source and the receiver are $6m$ apart and aligned in the angular direction. Their radial position is $r_c = 20mm$. As reported in section 4.1.1.3, the maximum group velocity in the frequency range of $x(t)$ is around $5km/s$. This velocity corresponds to an arrival delay of roughly $1.2ms$ at 6 meters of distance. The same time gap can be observed in figure 4.36.

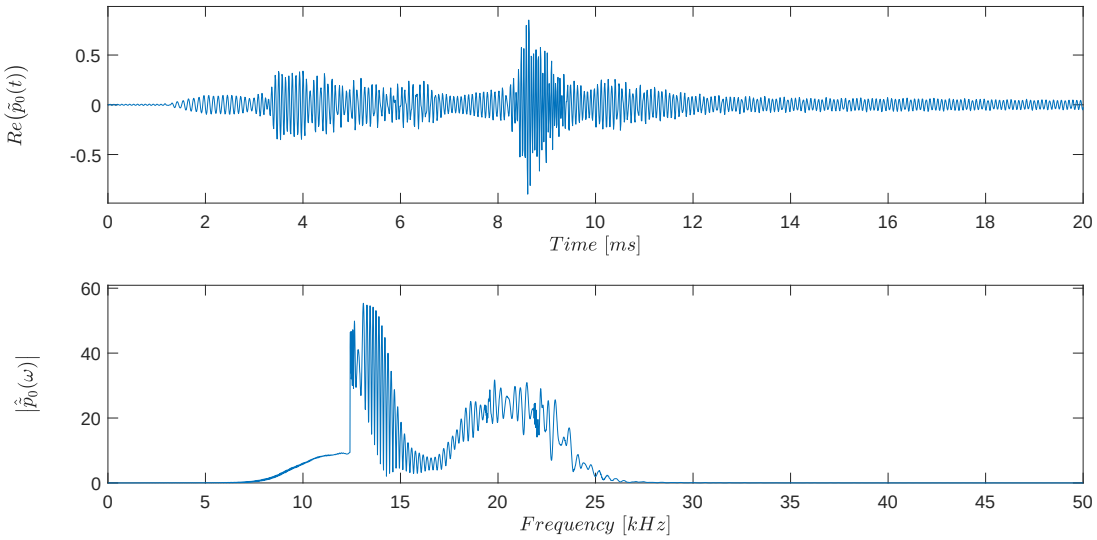


Figure 4.36: *Time and frequency domain representation of the dimensionless output pressure at the recording point G of the pipe illustrated in figure 4.1. Pipe material: aluminium,* $W_1 = 49.20mm$, $W_2 = 50.80mm$, $R_P = 0.0225m$, $C_P \equiv (0.02m, 0°, 0m)$, $G \equiv (0.02m, 0°, 6m)$.

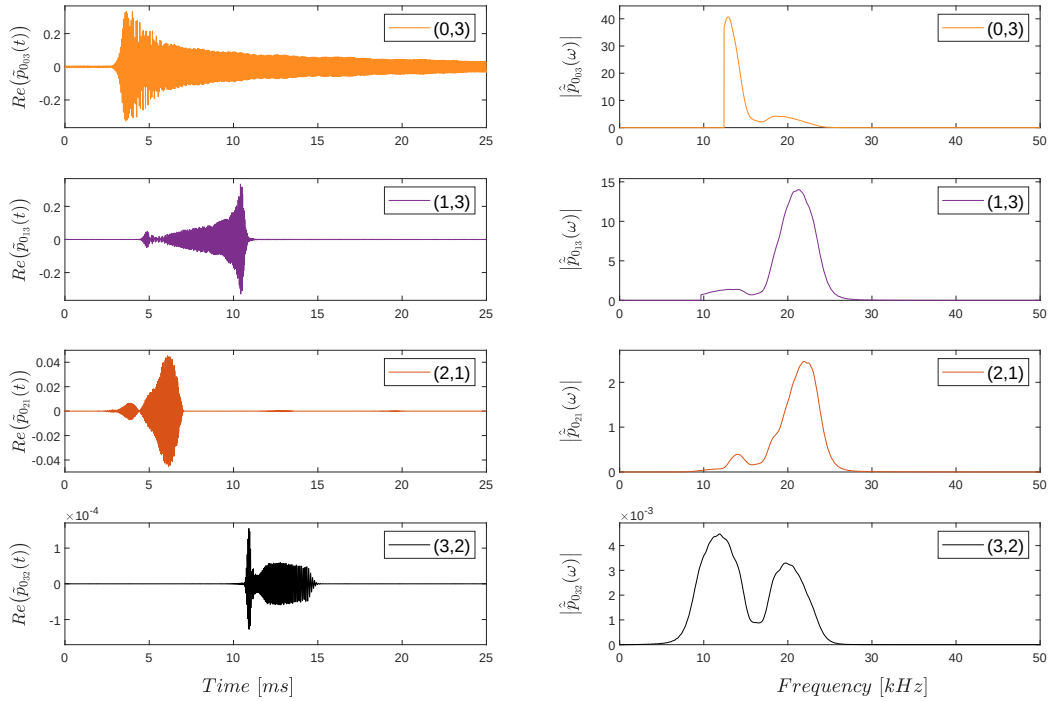


Figure 4.37: *Representation of the dimensionless pressure for the modes* $(0,3)$, $(1,3)$, $(2,1)$, $(3,2)$ *composing the signal reported in figure 4.36. Mode* $(0,3)$ *exhibits strong dispersive behaviour and discards the spectrum below its cut-on. Mode* $(2,1)$ *propagates at the shear velocity of the aluminium shield and is non-dispersive. Modes* $(1,3)$ *and* $(3,2)$ *are only slightly dispersive, the latter propagating below the speed of sound in water.*

The output signal described above is the summation of several modes with different characteristics. For some of them, the time and frequency domain representation of the dimensionless pressure is reported in figure 4.37. Mode $(0,3)$, for instance, exhibits a strong dispersive behaviour. The spectrum of the input signal spreads around its $12.421kHz$ cut-on, and all the frequency components in the evanescent region are not propagated. This issue appears as a sharp spectral step at the beginning of the propagative range. The frequency components just above the cut-on propagate at very low group velocity, causing a long tail lasting far beyond the duration of the input signal. Note that the maximum group velocity of the mode $(0,3)$ in the range considered is around $5km/s$ (figure 4.5). Nevertheless, this speed concerns only the last part of the chirp signal. The first part spreads below the cut-on or propagates at a very low group velocity. Consequently, the initial delay for mode $(0,3)$ is around $3ms$. Very different behaviour can be observed for mode $(2,1)$. In figures 4.9 and 4.11, it is shown that this mode is non-dispersive and propagates at the shear velocity of the aluminium shield ($v_g = 3040m/s$, initial delay $1.97ms$). Despite its non-dispersive characteristic, however, an evident deformation appears in both time and frequency representation. Indeed, this alteration is caused by the value of the coefficients obtained for the specific pressure source as described in section 4.3.

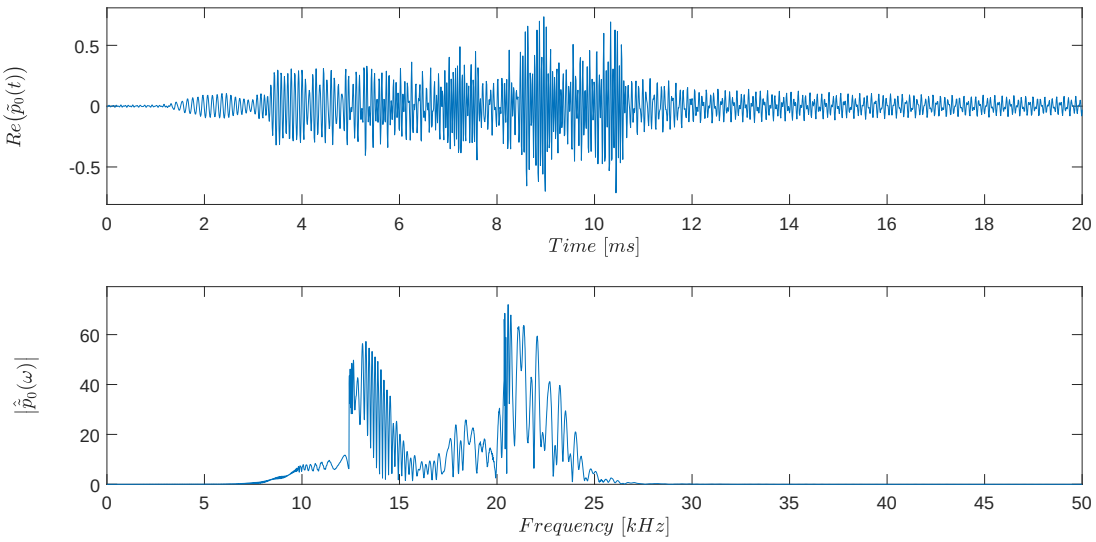


Figure 4.38: *Time and frequency domain representation of the dimensionless output pressure at the recording point G of the pipe illustrated in figure 4.1. Pipe material: aluminium,* $W_1 = 49.20mm$, $W_2 = 50.80mm$, $R_P = 0.0225m$, $C_P \equiv (0.02m, 0°, 0m)$, $G \equiv (0.02m, 45°, 6m)$.

Modes $(1,3)$ and $(3,2)$ are only slightly dispersive, and their duration is similar to the duration of the input signal. Mode $(3,2)$ propagates at the slowest group velocity and reaches the receiver after more than $10ms$. It roughly preserves the shape of the spectrum, but its contribution is tiny and completely masked by the tails of other dispersive modes. A similar example is reported in figure 4.38, where the angular position of the receiver has been rotated by $45°$, meaning $C_P \equiv (0.02m, 45°, 6m)$. Note how, although similar to the one reported in figure 4.36, the spectrum of the output signal is affected by the $e^{in\theta}$ dependence of the modes. In signal post-processing, this periodicity can be exploited to cancel or amplify modes by combining recordings obtained at different points around the axis of the waveguide.

A further meaningful example concerns the experimental measurement of the group velocity, as described in section 4.2.1. For a clear visualisation, the source and the receiver are now assumed in the central position at a distance of 6 meters. As already pointed out, only $n = 0$ modes should be considered in this case. Indeed, an axisymmetric source does not require any $e^{in\theta}$ variation, and since $J_n(0) = 0$ for $|n| > 0$, modes for $n \neq 0$ can be neglected when the receiver is placed at the central position. The driving signal represented in figure 4.39 is a tapered sine with its frequency band centred at $18kHz$.

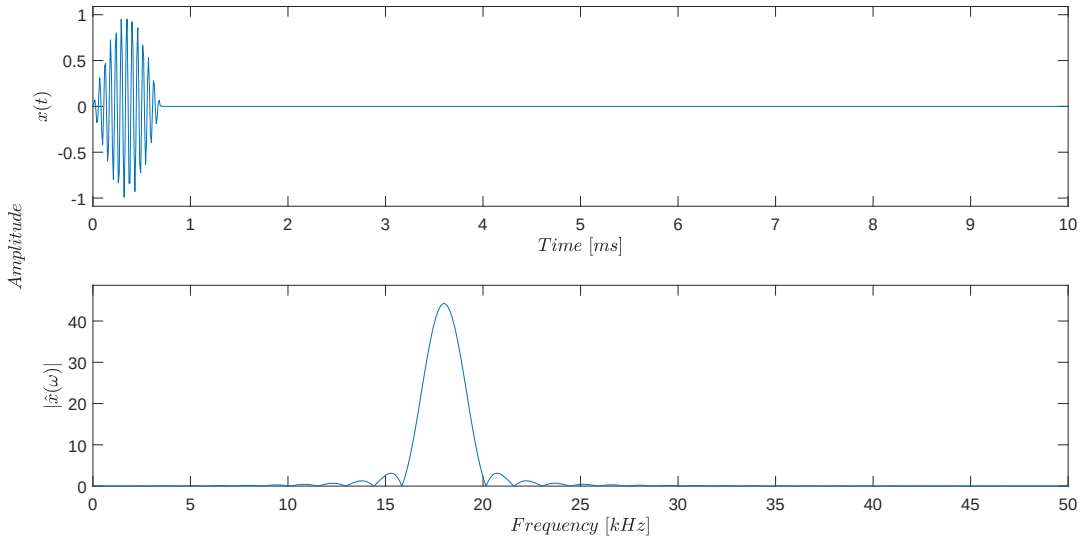


Figure 4.39: *Time and frequency domain representation of the dimensionless tapered sine used as the input signal for group velocity measurement. The signal is built to obtain a short time duration and a narrow frequency band.*

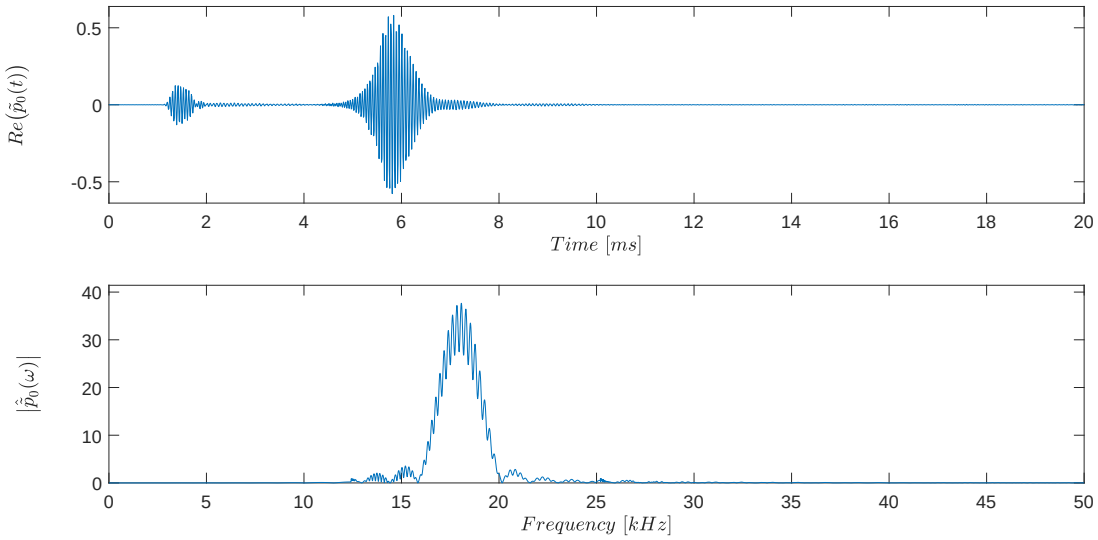


Figure 4.40: *Group velocity measurement. Time and frequency domain representation of the dimensionless output pressure at the recording point G. Pipe material: aluminium, $W_1 = 49.20mm$, $W_2 = 50.80mm$, $R_P = 0.0225m$. Source and receiver in axisymmetric position at a relative distance of $6m$. Two different modes propagating at different group velocities can be observed as separate entities in the output signal.*

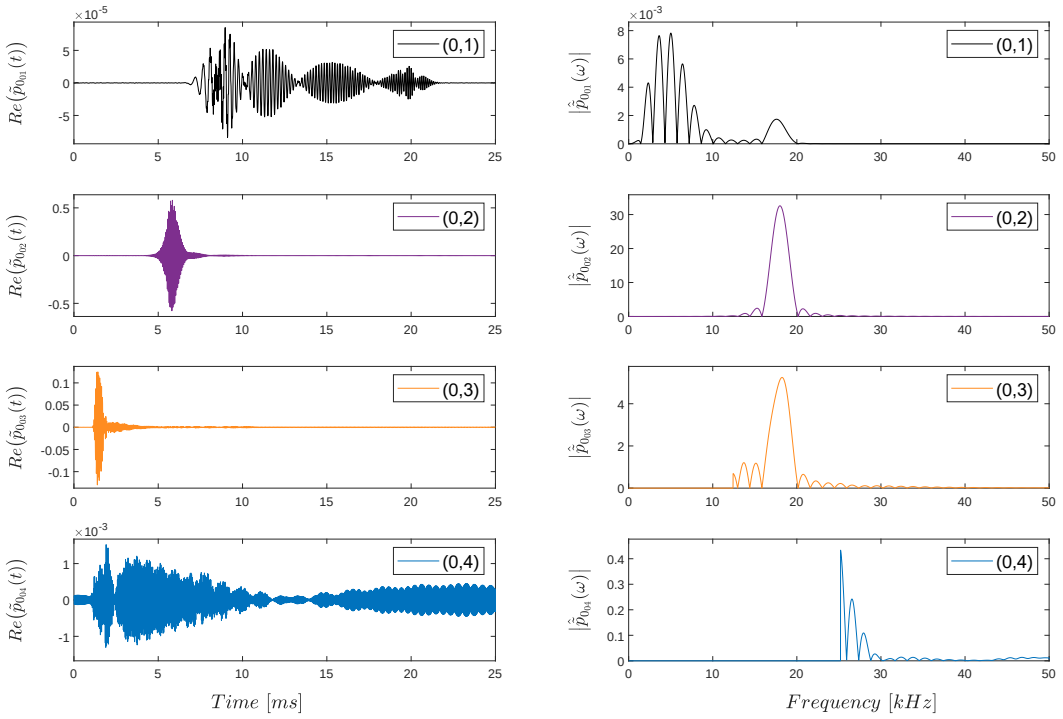


Figure 4.41: *Group velocity measurement. Dimensionless pressure time and frequency domain representation of the first four $n = 0$ modes for the output signal of figure 4.40. Given their amplitudes, only modes $(0,2)$ and $(0,3)$ are fully visible when all the modes are combined together. Modes $(0,1)$ and $(0,4)$ exhibit dispersive behaviour caused by the spectral tails of the input signal.*

The spectrum and the duration of the driving signals are chosen to satisfy (4.9), which guarantees a good separation between the train of pulses propagated by different modes and recorded in G. Figure 4.40 reports the output signal at the receiver. Note how two groups of pulses appear perfectly separated in time. This result is better understood by analysing the isolated modes illustrated in figure 4.41. According to figure 4.5, the group velocity of mode $(0,2)$ at $18kHz$ is slightly below $5km/s$, while mode $(0,3)$ propagates roughly at $1.2km/s$. At a distance of $6m$, these velocities correspond to delays of $1.2ms$ and $5ms$, respectively. These time gaps can be clearly observed in figures 4.40 and 4.41, and offer a direct solution for the measurement of the group velocities. Mode $(0,1)$, despite being in its propagative range, exhibits a tiny amplitude, and its detection in the output signal is difficult. Besides, the amplitudes of its coefficients strongly modify the spectrum, and the spectral tail of the input signal toward the lower frequencies is better preserved than the rest. This explains the evident dispersive behaviour shown in the top boxes of figure 4.41. Finally, the cut-on of mode $(0,4)$ $(25.208kHz)$ lies above the central frequency of the input signal. Nevertheless, the upper spectral tail of the latter extends above this boundary, which explains the tiny signal with the visible dispersive behaviour illustrated in the bottom boxes of figure 4.41.

4.5 Summary

This chapter presents the acoustic model developed in this work and the related elements of novelty. Starting from the equations reported in chapter 2, it introduces a quasi-analytical simulator organised in three layers and aimed to minimise the time required to calculate non-attenuated in-pipe reverberations. The first layer accounts for the materials and the geometry of the waveguide, yielding the dispersion curves for the modes along with the related phase and group velocities. Two novel algorithms are implemented for the calculation and the separation of the axial wavenumbers. In particular, the proposed solution for the latter provides a workaround for non-orthogonal modes, a case not reported in the literature but crucial for the present work. The effects of the variation of mechanical and geometrical properties are qualitatively discussed by comparing

the results for waveguides with different geometries and materials. The second layer accounts for the acoustic source, offering a solution for calculating the modal coefficients given a certain source pressure distribution. Two novel strategies are proposed to deal with the non-orthogonality hypothesis. In the first case, the algorithm aims to minimise the distance between the source pressure distribution and its approximation obtained from a limited number of modes. In the second case, modes are analytically decoupled by solving a specific matrix equation. Since the formulation of the problem is ill-conditioned, a further transformation based on the singular value decomposition is discussed to improve the accuracy of the results. An example of a pressure source modelled as a baffled piston is also considered for reference. Results show that, for both algorithms, the approximation error for a smooth axisymmetric source mainly remains below 1%, being the analytical decoupling approximately one order of magnitude better. A much larger number of modes is required for similar approximations of sources in non-axisymmetric positions. The last layer of the model calculates the desired reverberation, establishing a transfer function between a given dimensionless input signal and the output pressure at a point G along the waveguide. A procedure for the direct measurement of the most relevant quantities is further provided and compared against other options in the literature. A related application for the measurement of the group velocity is analysed in the last section. Finally, the developed model can be matched to real low-cost test rigs for practical in-house experiments and is used in chapter 5 for the acoustic data augmentation of the synthetic dataset proposed.

Chapter 5

Synthesis of augmented data

This chapter describes how synthetic augmented observations are generated in this work. The next section discusses the motivations underlying the adoption of artificial data and analyses similar solutions reported in the literature. Then, the details about the methodology used for the synthesis are reported. The first step concerns the creation of the soundbank and its structure. A brief overview of the hardware acquisition chain is given to understand the strengths and limitations of the adopted solutions. Section 5.3 proposes a statistical spectral noise filter to improve the quality of the synthesised audio and the related controlled properties. Section 5.4 introduces the synthesis and describes how audio blocks are combined to obtain new observations with properties and annotations as required. Finally, the acoustic model presented in chapter 4 is integrated to add simulated in-pipe reverberation. The processing chain developed in the following sections is employed in chapter 6 to generate the necessary observations for the creation of the synthetic dataset.

5.1 Motivations for data augmentation

In chapter 3, it is pointed out that every machine learning system requires one or more datasets to train the algorithms. Audio data archives for machine learning purposes have become more common in recent years, and several examples can be found online. Some of them, such as *Freesound* [247] and *Audioset* [248], are

organised as general purpose repositories, while others are structured datasets for specific purposes (e.g. *TUT* [39], *SONYC-UST* [40], *DESED* [249]). Frequently, the latter are extracted from the former, integrated with additional data and published as part of calls for machine learning challenges [250],[251]. This work focuses on in-pipe audio events and, in particular, on *domestic in-pipe audio events*. Practical reasons dictate the domestic choice, but the methodology can be extended to similar applications. To the best of our knowledge, no similar dataset is available in the literature.

Usually, apart from other acoustic specifications, good quality datasets for machine learning should be *strongly labelled*, meaning that -at least- class annotations and timing data are specified for all the events. Given the difficulties related to the creation of such repositories, *weakly labelled* datasets with labels given without timing data or just given at segment level are frequently provided [38, p. 154]. In general, depending on the requirements, collecting suitable recordings is difficult and time-consuming. For instance, for what concerns this work, direct acquisition of a reasonable number of audio examples would require unpractical access to a large number of different venues. As a consequence, it is difficult to provide enough variability and to cover additional acoustic effects that are typical of in-pipe propagation. Besides, the manual creation of strong annotations would be time-consuming and unavoidably affected by human inaccuracies.

Despite the differences, gathering large reliable datasets is a common barrier against the development of new AI algorithms and some solutions proposed aim to ease the problem by creating datasets of artificially generated sound scenes [252],[56]. *Scaper* [58], for instance, is a software package that can be used to generate sound observations by combining a set of isolated sounds in a *soundbank*. The underlying idea is to join a *background signal* together with multiple *foreground signals* representing the acoustic events to detect. Signal-to-noise ratio S/N and other acoustic properties can be assigned to control and diversify the occurrences of the chosen classes in the synthesised signals. Over the entire dataset generated, each parameter is controlled according to a set of probabilistic rules. Since the original classes are known, the synthesised recordings are generated together

with the related labels, and it is possible to obtain strongly labelled datasets that are also arbitrarily large.

Given the difficulties in the in-pipe scenario, a similar approach can be profitably exploited. However, it should be pointed out that in-pipe acoustic observations cannot always be synthesised by the simple superimposition of signal chunks from a soundbank. Indeed, sounds propagating in waveguides are strongly affected by reverberation, and the complexity increases if additional reflections or scattering are considered. Therefore, a simple cut and paste of parts of recorded signals would return unrealistic results and limited variability. Similar issues have been analysed in different contexts. Kinoshita et al. [57], for instance, investigated reverberation in the context of room acoustics for Automatic Speech Recognition. For in-pipe acoustics, although only for straight and circular pipes, the model proposed in chapter 4 provides a tool to calculate the above-mentioned effects. Besides, since a similar mechanism always regulates reverberation, the model maintains its usefulness despite its simplicity.

Accounting for the issues above, the adopted dataset is created by generating semi-synthetic observations obtained by isolating and recombining clean, non-reverberated audio blocks from real sources. Then, timing, S/N, reverberation, and other acoustic properties can be imposed, controlled, and accurately annotated.

5.2 The soundbank

In order to create the synthetic repository described above, a collection of low-reverberated sounds related to the desired class instances must be acquired. This collection of sounds is named *soundbank*, and this section describes the experimental procedure to acquire and organise its content. A description of the hardware setup and the related motivations is provided along with the procedure for the signal acquisition; a further section is about the structure of the soundbank. It is shown how the methodology proposed allows the minimisation of the human tasks required and how this is achieved by avoiding the direct collection of metadata. The last section introduces the *synthetic soundbank*, later detailed in section 5.4.1

5.2.1 Hardware acquisition chain

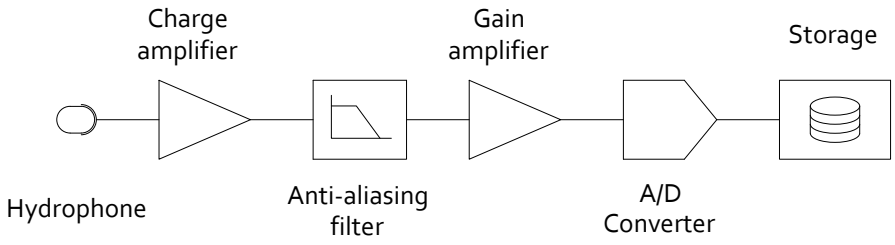


Figure 5.1: *Soundbank acquisition chain. The signal is acquired using a single hydrophone, conditioned, converted in samples of 16-bit, and stored in uncompressed digital format.*

Source signals are recorded using the acquisition chain illustrated in figure 5.1. The acquisition chain includes a hydrophone, a signal conditioning stage, an analogue-to-digital converter (A/D), and a storage unit. As better shown in chapter 6, for the signals recorded, most of the energy is concentrated below $4 \div 5kHz$ with some weaker components up to around $10kHz$. Since signals propagate in liquid and solid domains, a hydrophone is the simplest choice for the transducer. The selected device is a Brüel&Kjær type 8103 [253], which has dimensions compatible with the application and exhibits a charge sensitivity $(0.097pC/Pa)$ roughly constant $(+1dB, -1.5dB)$ in a range larger than the minimum required $(0.1 - 20000Hz)$. An extract of the datasheet concerning the hydrophone frequency response is reported in figure 5.2. The charge amplifier, the anti-aliasing filter, and the gain amplifier are part of the same Brüel&Kjær Nexus 2692 [254], which is the recommended coupled device for the hydrophone chosen. Output voltage and anti-aliasing frequency range are compatible with the following A/D sampling stage. The analogue-to-digital converter is a 16-bit resolution ADLINK USB-1210 [255], where the converter resolution is chosen considering the noise floor of the analogue chain. Indeed, a 24-bit conversion would require higher costs (e.g. equipment, processing, storage), while no relevant additional information would be stored in the least significant 8 bits. The whole acquisition chain is configured so

that the numeric values stored represent the actual value of the acoustic pressure in *Pascal*.

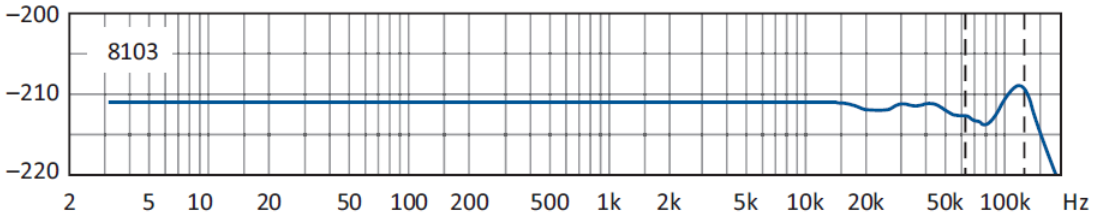


Figure 5.2: *Hydrophone frequency response (Brüel&Kjær type 8103 datasheet). Values reported in dB Re*$1V/\mu Pa$.

5.2.2 Acquisition of the soundbank signals

When recording audio items in their natural environment, noise and other undesired acoustic phenomena can affect or mask the actual sound emitted by the acoustic sources. In pipes, sounds propagate through complex geometries and layers of different materials, and acoustic distortions cannot be avoided. Nevertheless, optimisations of the recording process are possible. For instance, the receiver can be positioned in direct contact or in the closest proximity of the sound source to improve the signal-to-noise ratio and reduce reverberation effects. Generally, the hydrophone cannot be placed directly in water, and special adaptors need to be used to guarantee an acceptable acoustic path. Indeed, as seen in section 2.6.3, when the path between source and receiver includes layers of very different characteristic acoustic impedance, most of the transmitted power bounces back at the boundary. Therefore, a high-density gel has to be spread on the contacting surfaces of the adaptors to eliminate residual air layers and close the acoustic path between the source and the receiver.

Figure 5.3 shows the hydrophone and an aluminium adaptor for a $15mm$ copper pipe during a recording session. The surfaces in contact with the pipes and the hydrophone slot are prepared with high-density acoustic gel before being clamped using two cable ties. Several other plastic and metallic adaptors have been used on pipes of different diameters, such as $25mm$ and $40mm$.

Figure 5.3: *Example of a recording session for the acoustic emission of a tap using a* $15mm$ *pipe adaptor. Sound samples are recorded from the pipe just beneath the sink.*

Given the spectrum of the signals and the available options for the aliasing filter, during the recording sessions, the (low-pass) cut-off frequency is set to $30kHz$, while the corresponding sample frequency to a minimum of $60kHz$. In post-processing, however, all signals are resampled at the standard frequency of $48kHz$, and a further reduction is possible. The gain of the analogue stage is determined according to the intensity of the specific source so that the amplified analogue signal covers the entire dynamic range of the A/D converter. The resampled signals are then stored in `.wav` format. Since the `.wav` requires amplitude between -1 and $+1$, the signals are normalised to their maximum value, and the *normalisation factor* is stored in the `comment` field of the file metadata. The values of the acoustic pressure in Pascal can be recovered from the `.wav` file by multiplying the normalised numeric values by the normalisation factor.

The recorded observations are organised in *classes*, and a different folder is used per each class. A class represents a specific group of objects emitting noise.

In this thesis, the following eight classes related to *domestic environments* have been identified:

- `Taps`
- `Sinks`
- `Showers`
- `Toilets`
- `Washing Machines`
- `Dish Washers`
- `Disturbances*`
- `Backgrounds*`

Each class folder contains a folder per *class instance* (e.g.: `tap 1`, `tap 2`, etc.). Each class instance contains three separate observation folders: *events*, *disturbances* and *backgrounds*. All the observations in the same instance folder are recorded using the same hardware setup. A single event or disturbance folder includes only one recording with a certain background noise level. Background noise observations are recorded without any event or additional disturbance for statistical noise processing purposes. Since classes and class instances are stored separately, a weak annotation is intrinsically contained in the folder structure and the related names. This approach avoids any further manual collection of metadata. More details about the structure of the soundbank and the special classes, `disturbances` and `backgrounds`, are given in section 5.2.3.

Some of the classes identified refer to physical appliances that might not always be connected to the same pipes. Toilet drainage, for example, is obviously not connected together with tap pipes. Pipes can also have different sizes and exhibit different reverberation effects. Hence, it is unrealistic to imagine a single real IoT device trying to discriminate all the above-mentioned classes and the related events from a single listening point. This issue, however, is irrelevant since we only aim to provide a set of examples related to in-pipe events and to understand how they can be processed and organised in a dataset for machine learning applications.

5.2.3 Structure of the soundbank

The structure of the `Soundbank` can be represented as in figure 5.4. The `Soundbank` folder contains all the source signals used to generate the synthetic blocks (stored in the `Synthetic Soundbank` folder — section 5.2.4) and the final synthetic observations (stored in the `Synthetic Observations` folder — section 5.4). The `Soundbank` contains a folder per each class, including the two special classes `Backgrounds*` and `Disturbances*` and a further attribute file `Classes Attributes.txt`. Except for the special ones, each class stores a folder per each *class instance* (e.g. `Tap 0`, `Tap 1`, `Tap n`), meaning a specific appliance and the related acquisition hardware set-up. Each class instance contains the folders `Background Observations`, `Disturbance Observations`, and `Event Observations`. The first folder stores the isolated background noise signals used for both the post-processing noise reduction and the synthesis of the synthetic background. The `Disturbance Observations` folder includes a set of unlabelled events recorded using the same hardware configuration of the related class instance. For example, it may include voices, music or other unclassified domestic sounds. The last folder, `Event Observations`, contains a set of observations related to the specific instance, that is, a set of sounds produced by the specific appliance. As shown in figure 5.4, each observation for backgrounds, disturbances and events is stored in a different folder (`Backgrund 0, ..., Background n; Disturbance 0, ..., Disturbance n; Event 0, ..., Event n`). Each observation folder contains a further `Source` folder where the source data is stored in the same format as produced by the ADC.

Once the real signals have been imported into the `Soundbank`, an additional post-processing step must be performed to attenuate the noise level for events and disturbances. Per each class instance, the noise mask described in section 5.3.1 is obtained from the `Background observations`. Then, events and disturbances are processed to attenuate the undesired background noise. The purpose of post-process filtering is not to obtain noise-free synthetic observations but to better isolate events and disturbances. In the final synthesis, the desired background noise can be superimposed, allowing precise control of the S/N ratio, thus accounting for different recording conditions.

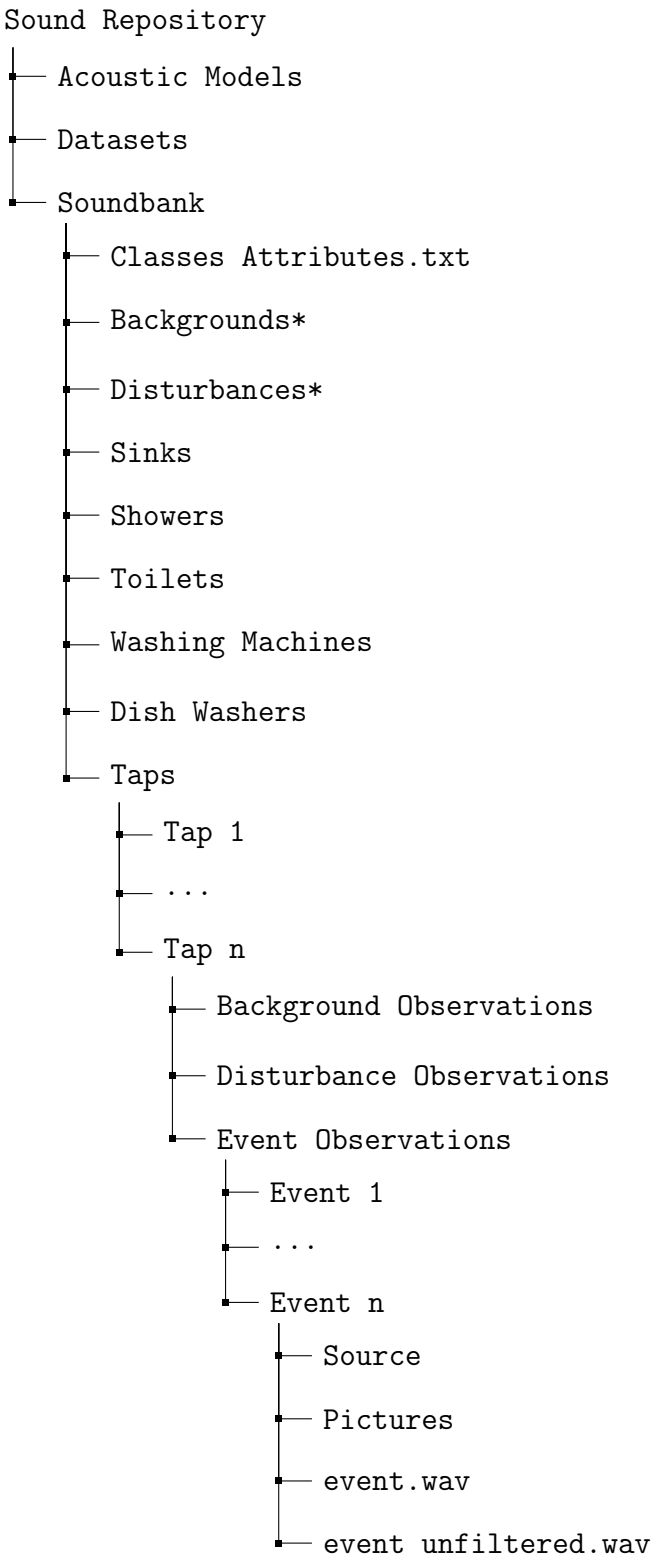


Figure 5.4: *Structure of the soundbank.*

Post-processed noise-free events are stored as `event.wav` in the corresponding `Event n` folder along with the unfiltered version of the event `event unfiltered.wav`. The amplitude is normalised and the scale factor to convert in pressure (*Pascal*) is stored in the `comment` field of the `.wav` file. The absolute position of the audio file in the soundbank is saved in the related `title` field of the file (e.g. `Soundbank > Taps > Tap 1 > Event Observations > Event 3`). A picture of the time-domain filter result is saved as `filter result.pdf` under the `Picture` folder.

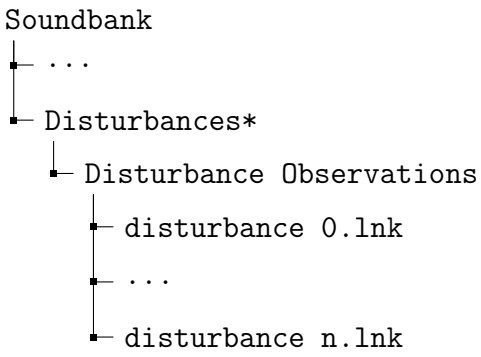


Figure 5.5: *Structure of the soundbank: special class Disturbances.*

Disturbance files are processed and organised as for the events, but a further step is performed in this case. Although disturbance recordings belong to specific class instances, for the synthesis of new observations, they are considered generic and not associated with any specific class instance. Hence, while their original collocation is maintained keeping the disturbance in the related folder, a link to the actual position is generated and stored under the folder `Disturbance Observations` of the class `Disturbance*` as shown in figure 5.5. Background observations are treated similarly to disturbances maintaining their original collocation in the class they belong to and generating a link under the `Background Observations` folder of the special class `Backgrounds*` (figure 5.6).

Finally, the file `Classes Attributes.txt` in the `Soundbank` folder defines a list of attributes used during the creation of the synthetic audio blocks. In particular, the attribute `include` defines which class can be used to generate a synthetic block,

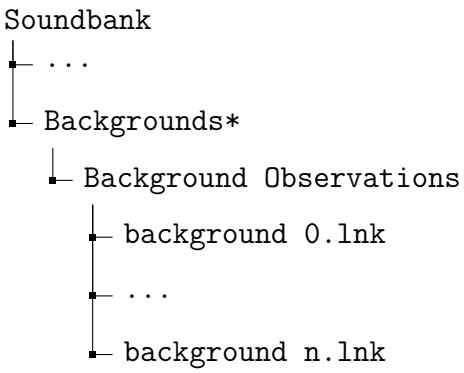


Figure 5.6: *Structure of the soundbank: special class Backgrounds.*

while the attribute `structured` indicates those classes whose source observations cannot always be considered homogeneous and should not be combined together when creating a new synthetic observation. More details about the creation of synthetic blocks are given in the section 5.4.

5.2.4 The synthetic soundbank

When a new synthetic dataset is created, the class/instance structure defined in the soundbank is replicated in a separate repository named `Synthetic Soundbank`.

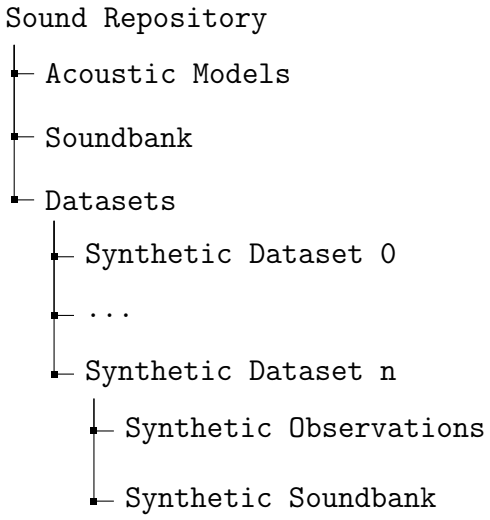


Figure 5.7: *Position of the synthetic soundbank.*

The content of the synthetic soundbank is not generated upfront during the post-processing stage of the soundbank but during the synthesis of the related dataset. Since the synthesis combines different blocks together, a copy of the isolated items is stored for reference and placed in the related class/instance folder of the synthetic soundbank. This solution allows the creation of multiple datasets while keeping track of the building audio blocks without affecting or replicating the source audio data. The position of the `Synthetic Soundbank` is reported in figure 5.7, while its content is described in section 5.4.1.

5.3 Noise reduction

The signal samples recorded to build the soundbank are affected by *noise* related to both the recording environment and the acquisition chain. Some non-exhaustive examples are the $50Hz$ harmonics coming from the power network and the quantisation noise related to the finite resolution of the analogue to digital converter.

The literature reports many noise reduction techniques, but the selection of a suitable algorithm depends on the specific context. In some cases, for example, methods based on *Wiener filters* or more advanced dynamic *noise cancellation* offer a remarkable solution to enhance the signal-to-noise ratio. These approaches, however, can be exploited only when additional information is given. Wiener filters, for example, require some preliminary knowledge about the nature of the noise, such as autocorrelation and cross-correlation [53, p. 339]. Similarly, noise cancellation algorithms must be fed with additional signals, which can be, for instance, a measure of the error or a correlated version of the disturbance [53, p. 349],[256]. Another possible approach is based on Kalman filters, which require modelling the observation as the output of a linear system, including the estimation of both system dynamics and system order [53, p. 371],[257].

For this work, an approach based on *spectral subtraction* or *noise gating* [258, p. 136],[259],[260] seems to be more suitable, meaning that good filtering performances are achievable without additional acquisition constraints and with low

computational complexity. Fast algorithms are also convenient for batch processing and real-time signal enhancement in devices with limited hardware resources. Two open-source implementation examples are the free software *Audacity* [261] and the python library *Noise reduce* developed by Sainburg et al. for post-processing of animal vocal repertoires [52]. The latter is used for performance reference in the following sections since it is recent, still maintained, and closer to our scope. The implementation here proposed aims to:

- improve the overall S/N gain achievable,
- eliminate filtering sound artefacts,
- provide an algorithm suitable for real-time and batch processing,
- provide an accurate methodology to assess the algorithm performance against different solutions.

The filter algorithm articulates in three parts: the first step concerns the determination of a statistical *noise filter mask* (section 5.3.1), the second step regards the synthesis of the filter (section 5.3.2), and the third step provides the actual signal filtering (section 5.3.3). A further software utility (section 5.3.4) has been developed for performance assessment.

5.3.1 Synthesis of the noise mask

Although stationary assumptions do not hold for the acquired recordings, under the hypothesis that the hardware setup remains unaltered during the acquisition of a class instance, the background noise can be considered mainly time-invariant. To obtain a robust characterisation, a statistical approach is used and a given set of pure noise recordings is divided into a *characterisation set* $\{Q\}$ and a *check set* $\{C\}$. The noise sets $\{Q\}$ and $\{C\}$ can be chosen to be disjoint or overlapping, the latter option being necessary when only one or a small number of noise recordings is available. Each recording in $\{Q\}$ is short-time Fourier transformed using windows of length L. A Hann windowing function is applied to smooth the time boundaries and reduce the frequency artefacts when transformed in the frequency domain. The *power spectral density (PSD)* $P_i[n_f]$ is derived $\forall\, i \in [0 : W_Q - 1]$ windows in $\{Q\}$. Then, the PSD noise mean $N_0[n_f]$, the PSD max

$N_{max}[n_f]$, and the standard deviation $\sigma_N[n_f]$ are calculated per each frequency bin $f[n_f]$. Finally, a noise mask $M_N[n_f]$ of length $L/2+1$ is found as the mean value plus the standard deviation times an *excess factor* $E_{\%}$:

$$M_N[n_f] = N_0[n_f] + E_{\%}\sigma_N[n_f] \quad with \quad \sigma_N[n_f] = \sqrt{\frac{\sum_{i=0}^{W_Q-1}\left(P_i[n_f] - N_0[n_f]\right)^2}{W_Q}}. \tag{5.1}$$

The desired excess factor is calculated by assigning a maximum percentage *excess threshold* $E_{T\%}$, that is, specifying the percentage of frequency bins of the check set $\{C\}$ that are allowed to be above the mask level. This solution enables precise control of the synthesised mask by using a direct specification on the outliers. A good choice for the excess threshold is a trade-off between filtering noise and preserving useful signal components that lay very close to the noise level. A range between 0.5% and 3% is a reasonable choice, being $E_{T\%} = 1\%$ the value assumed as default. The determination of the $E_{\%}$ by imposing an excess threshold on the noise check-set differentiates the calculation of the noise mask from the other noise gating approaches reported above [261],[52]. Note that some of the recordings used for synthesising the noise mask may be more similar to the actual noise to filter than others. From this point of view, assessing the mask on a single noise recording from $\{Q\}$ can occasionally offer a more accurate synthesis, but the selection of the best option cannot be controlled. Algorithm 3 reports the pseudo-code for the synthesis of the noise mask M_N given the input noise recordings $\{N_{rec}\}$ and the excess threshold $E_{T\%}$.

Figure 5.8 (top) illustrates an example of mask $M_N[n_f]$ with $E_{\%} = 1.6$ obtained by imposing $E_{T\%} = 2\%$ (green line). The spectrum ranges from 0 to $24kHz$ (sample rate $48kHz$). Time windows are $50ms$ long, giving 1201 frequency bins $f[n_f]$ including null and Nyquist frequency. Each frame is smoothed with a Hann function and overlaps the two adjacent windows by 50%. The yellow line reports the mean noise level $N_0[n_f]$, while the red line represents the max noise values $N_{max}[n_f]$. Finally, the blue line reports the power spectral density of a window that includes some useful signals. Values in dB are referred to the mean of N_0.

Algorithm 3 Synthesis of the noise mask

Input: $\{N_{rec}\}, E_{T\%}$

Output: M_N, N_0, N_{max}

1: $\{Q\}, \{C\} \leftarrow from\ \{N_{rec}\}$ ▷ Obtain noise sets

2: $N_0, N_{max}, \sigma_N \leftarrow from\ \{Q\}$ ▷ Calculate statistics

3: $M_N \leftarrow N_0,\ k \leftarrow 0$ ▷ Init noise mask

4: **while** $(\{C\} > M_N)_{\%} > E_{T\%}$ **do** ▷ Check excess threshold

5: $\quad k \leftarrow k+1$

6: $\quad E_{\%} \leftarrow k * \Delta E_{\%}$ ▷ Update excess factor

7: $\quad M_N \leftarrow N_0 + E_{\%}\sigma_N$ ▷ Update mask

8: **end while**

9: **return** (M_N, N_0, N_{max})

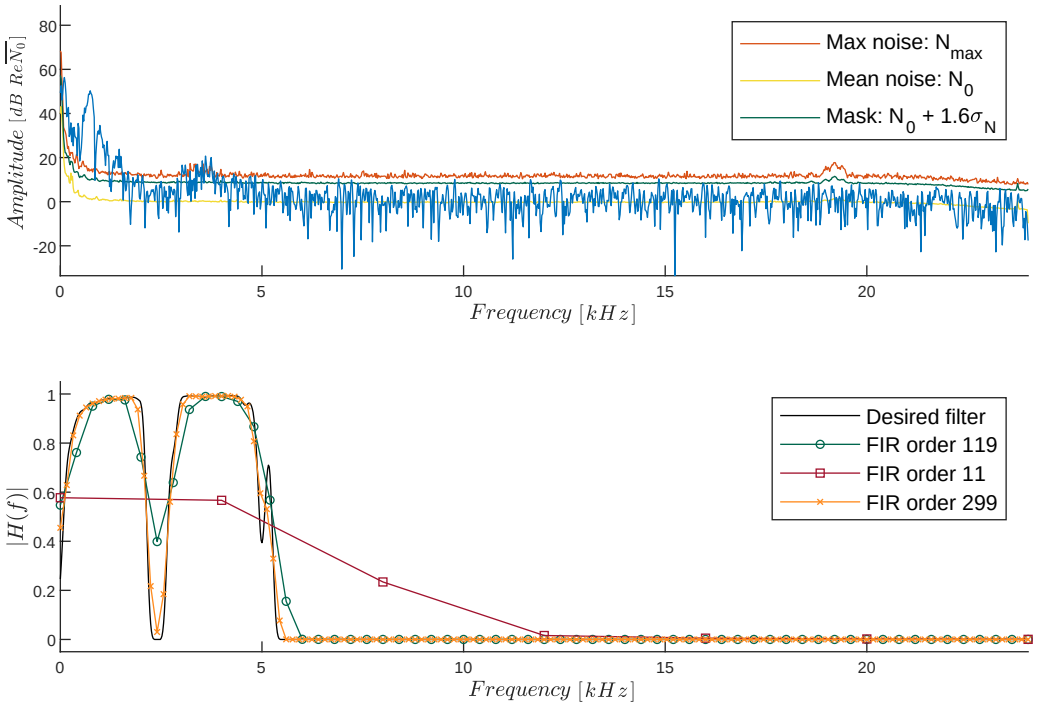


Figure 5.8: *Filter modelling. Window length* 50*ms, window overlap* 50%*. (Top) - Noise filter mask obtained with* $E_{\%} = 1.6$*,* $E_{T\%} = 2\%$*. Noise mask* M_N *(green line), noise floor* N_0 *(yellow line), max noise* N_{max} *(red line). The power spectral density of a window including a useful signal is represented in blue. Values in dB are referred to the mean of* N_0*. (Bottom) - Filter implementation using FIR filters of different orders. Frequency density radius* $R_{FD} = 100Hz$*, time density radius* $R_{TD} = 200ms$*, frequency smooth radius* $R_{FS} = 150Hz$*, time smooth radius* $R_{TS} = 125ms$*.*

5.3.2 Filter synthesis

The filter mask obtained in the previous section is employed to model the actual noise filter. In [52] the signal PSD is directly compared to the noise mask, gated, and the resulting boolean mask is frequency and time smoothed. Smoothing the filter mask in frequency and time undoubtedly attenuates sparse noise components above the level of the noise mask. The magnitude of this reduction, however, must be determined as a trade-off between reduction of noise and loss of filter selectivity, which translates into stronger attenuation of the useful signal and worse attenuation of the other noise components.

Accounting for this issue, the solution proposed aims to suppress the bins with random noisy behaviour by identifying the components that appear isolated either in frequency or in time. This approach provides a double benefit: it directly reduces the noise in the recordings and allows the synthesis of more selective filters by relaxing the requirements for time and frequency smoothing. The only drawback relates to the possibility of processing signals with very short or narrow-band features, a scenario safely negligible in the context of this work.

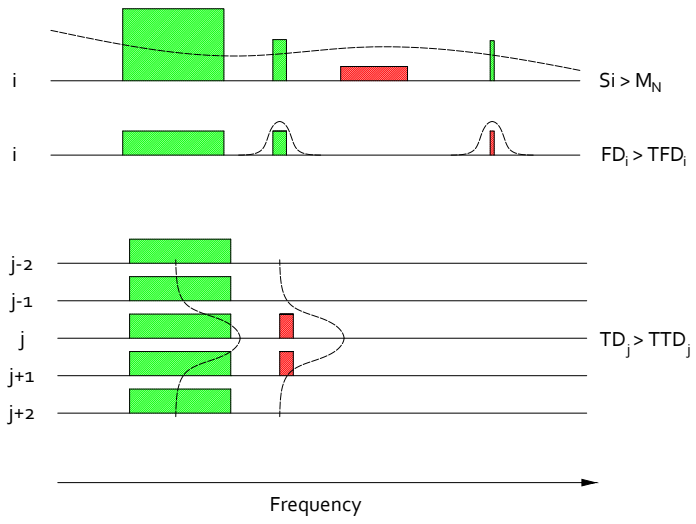


Figure 5.9: *Filter density masks. (Top) - The PSD of i^{th} window S_i is first compared against the noise mask M_N to eliminate the components below the noise floor. Then, the weighted sum of the frequency neighbours is compared against the frequency density threshold TFD_i. (Bottom) - The time density for the window j is evaluated over multiple neighbouring windows. Spectral time neighbours are weighted before being compared against the time density threshold TTD_j. Blocked components are represented in red.*

Algorithm 4 Synthesis of the k^{th} filter mask

Input:

$S_i, S_j, M_N, N_0, N_{max},$

$\boldsymbol{FD}, \boldsymbol{TD}$

$TFD_{min}, TFD_{max}, TTD_{min}, TTD_{max}$

$R_{FD}, R_{TD}, R_{FS}, R_{TS}$

Output: $|H_k|$

$i \leftarrow k + R_{TD} + R_{TS}$ ▷ Index of the new input window

$\alpha_i = [min(S_i - M_N)_{10\%}/10dB]_0^1$ ▷ Frequency factor

$TFD_i \leftarrow TFD_{min} + \alpha_i \times (TFD_{max} - TFD_{min})$ ▷ Update TFD_i

if $S_i < N_{max}$ **then**

  $FD_i \leftarrow 0$ ▷ Null mask

else

  $FD_i \leftarrow S_i > M_N$ ▷ S_i thresholding

  $FD_i \leftarrow gaussSum(FD_i, R_{FD})$

    ▷ Frequency neighbours weighted sum of radius R_{FD} for the i^{th} window

  $FD_i \leftarrow FD_i > TFD_i$ ▷ FD_i thresholding

  $\boldsymbol{FD} \leftarrow [\boldsymbol{FD}_{(2:end)}, FD_i]$ ▷ Discard the oldest, save new FD_i

end if

$j \leftarrow k + R_{TS}$ ▷ Time density mask index

$\beta_j = [min(S_j - M_N)_{10\%}/10dB]_0^1$ ▷ Time factor

$TTD_j \leftarrow TTD_{min} + \beta_j \times (TTD_{max} - TTD_{min})$ ▷ Update TTD_j

$TD_j \leftarrow gaussSum(\boldsymbol{FD}_{(j-R_{TD}:j+R_{TD})}, R_{TD})$

    ▷ Time neighbours weighted sum of radius R_{TD} for the j^{th} window

$TD_j \leftarrow TD_j > TTD_j$

$TD_j \leftarrow TD_j \times \left(1 - \frac{M_N - mean(M_N)}{max(M_N) - mean(N_0)}\right)$ ▷ Scale TD_j

$TD_j \leftarrow gaussSmooth(TD_j, R_{FS})$ ▷ Smooth TD_j in frequency with radius R_{FS}

$TD_j \leftarrow TD_j / max(TD_j)$ ▷ Normalise TD_j

$\boldsymbol{TD} \leftarrow [\boldsymbol{TD}_{(2:end)}, TD_j]$ ▷ Discard the oldest, save TD_j

$|H_k| \leftarrow gaussSmooth(\boldsymbol{TD}_{(k-R_{TS}:k+R_{TS})}, R_{TS})$

    ▷ Smooth TD_k in time with radius R_{TS}

return $|H_k|$

To build the filter mask, a *frequency density mask* $FD_i[n_f]$ is first obtained from the PSD of the i^{th} window of the input signal $S_i[n_f]$ (figure 5.9 - top). $FD_i[n_f]$ is assumed null if all the components of S_i are below $N_{max}[n_f]$. In the other cases, the bins above the noise mask $M_N[n_f]$ are initially set to 1. The importance of the frequency neighbours is weighed using Gaussian windows of radius R_{FD}, meaning higher weights for closer neighbours. Then, the weighted values of all the neighbours are summed and divided by the sum of the samples of the Gaussian window ($gaussSum$ in algorithm 4). Finally, the frequency density mask $FD_i[n_f]$ is obtained by setting to 1 the bins above the *frequency density threshold* TFD_i and leaving null the rest. This solution discards isolated noisy components and allows clusters of frequency bins. A similar operation is repeated in time (figure 5.9 - bottom). The last frequency density mask (step i) is buffered in $\boldsymbol{FD}$ discarding the oldest. This buffering shift centers $\boldsymbol{FD}$ around the j^{th} window, and the time neighbours of the latter can be weighted using a Gaussian function. The *time density mask* $TD_j[n_f]$ is obtained by setting to 1 only those bins for which the sum of the weighted time neighbours divided by the sum of the samples of the Gaussian window of radius R_{TD} is greater than the *time density threshold* TTD_j.

Frequency and time density thresholds TFD_i and TTD_i are defined in a given range ($[TFD_{min}, TFD_{max}][TTD_{min}, TTD_{max}]$) and adapted dynamically, being higher when the signal level is higher above the noise level. In particular, the proportionality factors (α and β in algorithm 4) are determined by measuring the level above the noise floor of the 10% highest samples of S_i and S_j, the input signal PSD for the windows i and j respectively.

The $TD_j[n_f]$ is then scaled according to the noise floor, making it lower where the noise floor is higher. Finally, the mask is frequency smoothed by convolution using a Gaussian window of radius R_{FS}[1] and normalised in the interval $[0, 1]$. The j^{th} mask processed as above is then saved in $\boldsymbol{TD}$, centering the buffer around the k^{th} window. Finally, the k^{th} mask is smoothed in time to obtain the final full-resolution filter mask $|H_k[n_f]|$. The time-smoothing window is Gaussian of radius R_{TS}. Note that a full-resolution filter mask is generally unnecessary and,

[1] Gaussian smooth is obtained using the *Matlab* function `smoothdata`

within a certain limit, masks can be down-sampled without affecting the final result. More considerations about this point are reported in sections 5.3.3 and 5.3.4.

It is remarked that processing the time density mask $TD_j[n_f]$ and the final time-smoothing introduces a temporal shift between a new window $S_i[n_f]$ and the output of a new filter mask $|H_k[n_f]|$. This delay, which is equal to $R_{TD}+R_{TS}$, should be kept low for real-time applications. The key features of the algorithm described are schematically given by the pseudo-code reported in algorithm 4.

Figure 5.8 (bottom), represents the desired filter $|H_k[n_f]|$ for three different sub-sampled solutions. The duration of the window is a trade-off between frequency and time resolution (section 3.2.1.1) [80, p. 89]. As for many sound processing applications [38, p. 21], a window of $20 \div 60ms$ provides a good balance. The frequency density radius R_{FD} applied is $100Hz$, the time density radius R_{TD} is $200ms$. Gaussian frequency smooth radius R_{FS} and Gaussian time smooth radius R_{TS} are $150Hz$ and $125ms$, respectively.

5.3.3 Signal filtering

To obtain the filtered signal, it is possible to apply the filter in either the time or the frequency domain. In the first case, the time-domain input signal is linear convoluted with the time-domain impulse response of the filter $h_k[n_t]$. In the frequency domain, each frequency component of the signal is multiplied by the correspondent component of $\hat{h}_k[n_f] = H_k[n_f]$ and the obtained spectrum is anti-transformed in the time domain. Sainburg [52], for instance, filters in the frequency domain after smoothing time-adjacent filters using a Gaussian function.

In the filtering process, a relevant issue concerns the introduction of undesired *sound artefacts*, that is, the alteration of the samples' expected value. This problem relates to both the windowing division and the filtering itself. When the input signal $x[n_t]$ is split into short overlapping windows $x_k[n_t]$ of length L, hop Q, and smoothing windowing function w, as long as the windowing functions satisfy the *non-zero overlap add (NOLA)* condition ($\sum_k \phi_k[n_t] = \sum_k \phi[n_t - kQ] \neq 0$), regardless the chosen hop, the original signal can be rebuilt from its windows.

Indeed:

$$\frac{\sum_{k=-\infty}^{+\infty} x_k[n_t - kQ]}{\sum_{k=-\infty}^{+\infty} \phi[n_t - kQ]} = \frac{\sum_{k=-\infty}^{+\infty} x[n_t]\phi[n_t - kQ]}{\sum_{k=-\infty}^{+\infty} \phi[n_t - kQ]} = x[n_t]\frac{\sum_{k=-\infty}^{+\infty} \phi_k[n_t]}{\sum_{k=-\infty}^{+\infty} \phi_k[n_t]} = x[n_t]. \quad (5.2)$$

When the windows $x_k[n_t]$ are processed using a filter $h[n_t]$, however, summing the filtered windows and dividing by the sum of the windowing function does not always return the desired filtered signal, meaning the signal obtained by filtering directly $x[n_t]$ with $h[n_t]$. A sufficient condition to obtain a correct reconstruction is:

$$\sum_k \phi_k[n_t] = C, \quad (5.3)$$

which is known as *constant overlap add* (*COLA*) condition. Besides, since C is a constant, it can be assumed unitary without loss of generality. To prove the statement above, it is recalled that, in the time domain, the filtered window $y_k[n_t]$ can be obtained by linear convolution:

$$y_k[n_t] = \sum_{j_t=-\infty}^{+\infty} h[j_t]x_k[n_t - j_t]. \quad (5.4)$$

Summing the filtered windows yields:

$$\begin{aligned}\sum_{k=-\infty}^{+\infty} y_k[n_t - kQ] = \sum_{k=-\infty}^{+\infty}\sum_{j_t=-\infty}^{+\infty} h[j_t]x_k[n_t - kQ - j_t] = \\ \sum_{j_t=-\infty}^{+\infty} h[j_t]\sum_{k=-\infty}^{+\infty} x_k[n_t - kQ - j_t] = \\ \sum_{j_t=-\infty}^{+\infty} h[j_t]x[n_t - j_t]\sum_{k=-\infty}^{+\infty}\phi[n_t - kQ - j_t],\end{aligned} \quad (5.5)$$

which, if the COLA condition with $C = 1$ is valid, returns:

$$y[n_t] = \sum_{j_t=-\infty}^{+\infty} h[j_t]x[n_t - j_t] = \sum_{k=-\infty}^{+\infty} y_k[n_t - kQ], \quad (5.6)$$

proving that summing the filtered windows returns the exact reconstruction of the filtered signal. Certain windowing functions, such as the Hann window, naturally satisfy the COLA condition for a specific overlap value (50% for the Hann). Although choosing the COLA exact overlap is the best way to preserve the function properties, the implemented noise filter allows an overlapping range between 25% and 50% by complementing the windowing function. Figure 5.10 illustrates the alteration of the Hann function aimed to reestablish the COLA condition for overlap values different from 50%.

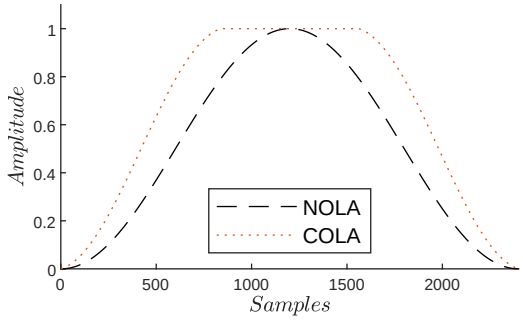


Figure 5.10: *Alteration of the Hann windowing function to reestablish the COLA condition. Overlap: 35%*

In a practical implementation, when the (5.4) is calculated using a *FIR*[2] filter $h_k[n_t]$ of order F, the sum can be limited to $F+1$ terms and, for an input signal of length L, equation (5.4) returns $L+F$ samples:

$$y_k[n_t] = \sum_{j=0}^{F} h_k[j_t] x_k[n_t - j_t]. \tag{5.7}$$

The output sequence exhibits a leading and a trailing interval (both of length F) where the convolution is calculated over a reduced set of points, and the filter response appears altered.

[2]Finite Impulse Response (FIR) filters are digital filters defined by an impulse response with a finite number of samples. On the contrary, the impulse response of an Infinite Impulse Response (IIR) filter exhibits an infinite number of non-null samples generated by some kind of feedback mechanism. In this work, only FIR filters are considered since the masks are obtained as a set of samples of finite length, and the conversion into FIR filters is immediate. Besides, FIR filters are always stable, and this avoids further complications during the synthesis.

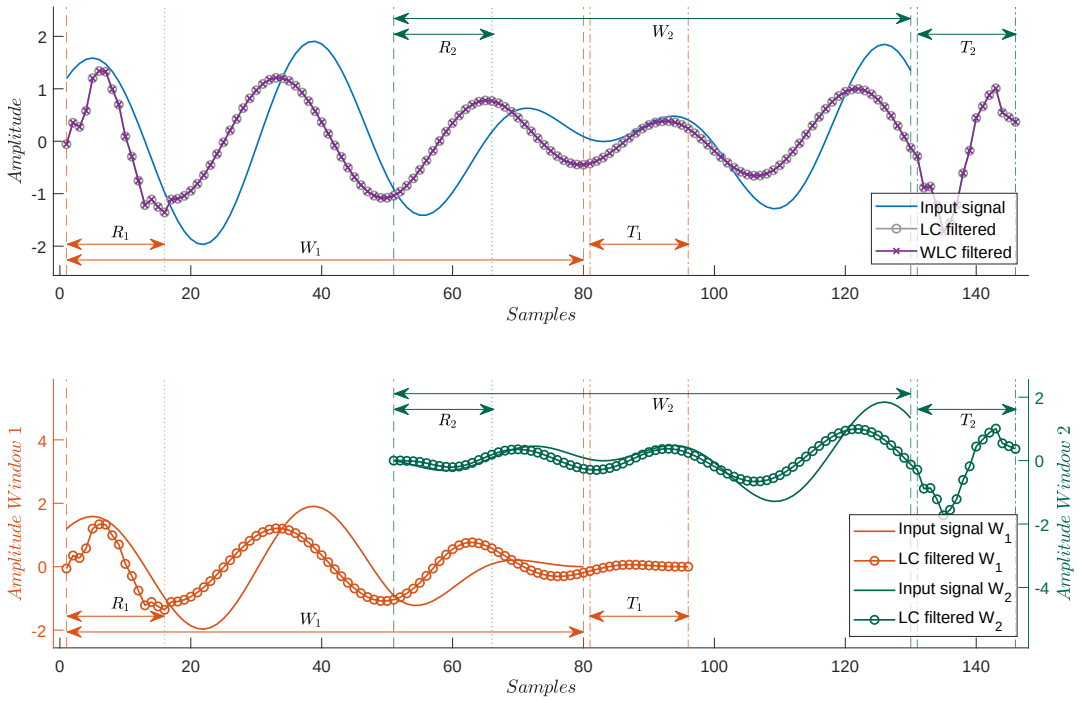


Figure 5.11: *Linear convolution. (Top) - Input signal (blue). Filtered output by linear convolution on the entire input signal (grey) and as the superimposition of the filtered windows W_1 and W_2 (violet). No artefact appears in the central part of the signal. (Bottom) - Input and filtered windowed signals. Leading R_i and trailing T_i artefacts are shown at the edges of the windows.*

In figure 5.11, a 130 samples long input signal $x[n_t]$ is divided by two trapezoidal overlapping COLA windows (W_1 and W_2) of length $L = 80$ and filtered using a *FIR* filter of order $F = 16$ ($h[n_t] = h_1[n_t] = h_2[n_t]$). In the bottom picture, the windowed signals are separated and leading (R_1) and trailing (T_2) artefacts are clearly distinguishable. Because of the windowing functions, artefacts T_1 and R_2 appear less pronounced. Since the filters $h_1[n_t]$ and $h_2[n_t]$ are the same, T_1 and R_2 are perfectly complementary, and the resulting overlapped signal is identical to the one calculated by processing without any windowing division (figure 5.11 top). When different filters are applied to adjacent windows, the artefacts are not perfectly complementary. From this point of view, increasing similarities between adjacent filters, as seen in the previous section with the Gaussian time smoothing, keeps the artefacts low.

When filtering is executed in the frequency domain, the output signal is generally calculated using the inverse discrete Fourier transform:

$$y_k[n_t] = \mathscr{F}^{-1}\Big(\hat{h}_k[n_f]\hat{x}_k[n_f]\Big). \tag{5.8}$$

This operation, however, is a *modulo-L circular convolution* [53, p.20] and, in general, it returns a sequence of L points that is different from (5.7). Circular convolution can be considered a linear convolution where the same input sequence of length L is repeated periodically. In each period L, the trailing tail wraps around and overlaps the leading tail at the beginning of the window. Hence, the returned output is correct only if the input signal is truly L periodic. If the circular convolution is calculated using a FIR filter of order $L-1$ with L samples, all the points of the output window are affected by sound artefacts.

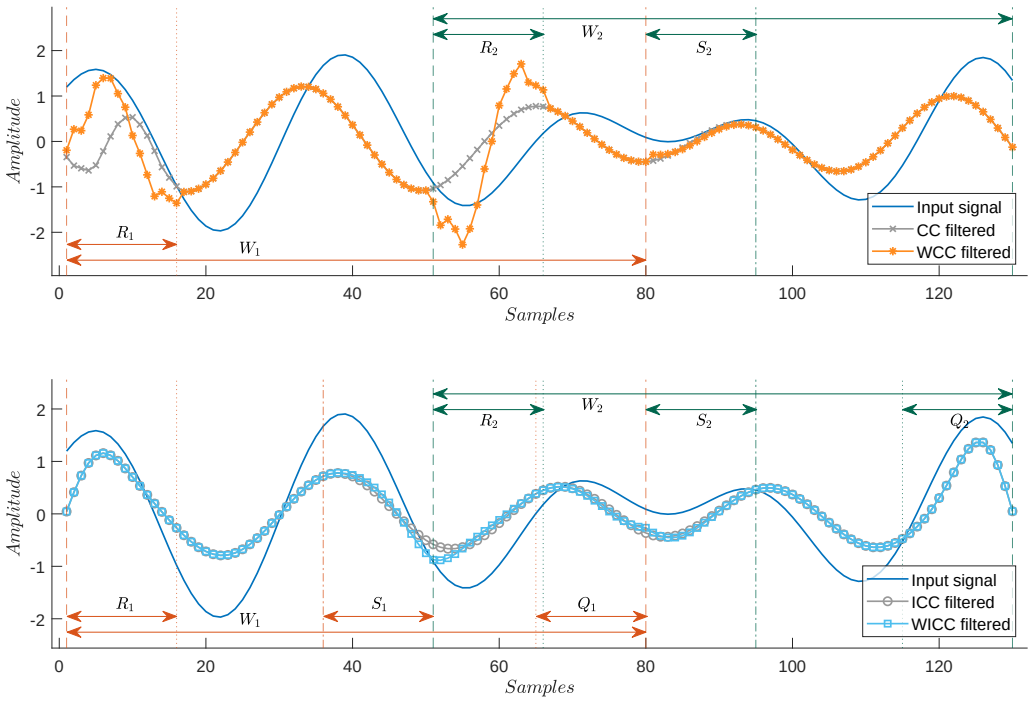


Figure 5.12: *Circular convolutions. (Top) - Input signal (blue). Filtered output by circular convolution on the entire input signal (grey) and as the superimposition of the filtered windows W_1 and W_2 (orange). Leading R_i artefacts are shown at the beginning of the windows. A further small artefact in S_2 appears just after the overlapping region. (Bottom) - Input signal (blue). Filtered output by improved circular convolution (Matlab `filtfilt`) on the entire input signal (grey) and as the superimposition of the filtered windows W_1 and W_2 (azure). Artefacts appear at the end and at the beginning of the windows since the signal is here processed in the forward and backward direction.*

To make the circular convolution equivalent to (5.7), it is necessary to pad the input signal and the filter impulse response with zeros up to the length $L+F$. This allows the trailing tail to develop without wrapping around and returns $L+F$ samples as for the linear convolution. Therefore, if filtering is performed in the frequency domain applying 5.8 with a filter of length L, artefacts can be avoided by padding the input signal $x_k[n_t]$ up to the length $2L$ and increasing the number of samples of the filters (by anti-transforming in the time domain, padding the impulse response up to $2L$, and transforming back in the frequency domain). This operation, however, increases the computational cost of filtering in the frequency domain.

The same example of 5.11 is repeated in figure 5.12 (top), where the windows are processed by circular convolution. Artefacts can be compared to the desired time-domain response. As shown in regions R_1 and R_2, artefacts are limited to the beginning of the windows (signal is processed only if forward direction), but the superimposition of the windows does not restore the desired output. Moreover, since the transient effects of the windowed smoothing function extend beyond the overlapping region, additional artefacts can appear just after the latter (S_2). As reported by Sadovsky [262] and implemented in *Matlab*[3], circular convolution artefacts can be reduced by solving a set of equations. The obtained result is closer to the desired one but requires higher complexity (figure 5.12 bottom).

5.3.4 Filter performance

Filter performance regards different issues. Computational complexity and filtering quality are analysed in the following subsections.

5.3.4.1 Computational complexity

Regarding the computational complexity, accounting only for the core operations, processing in the time domain is faster only when the filter order is small compared

[3] *Matlab* function `filtfilt` provides zero phase shift by processing the input data in both forward and reverse directions and corrects the artefacts at the edges of the filtered window. A phase shift correction is unnecessary when the filter frequency spectrum is purely real.

to the size of the window. Assuming the fastest algorithm [263], the *computational complexity order of approximation* of the *fast Fourier transform (FFT)* is $O(Nlog_2N)$ and, for a window, filtering in frequency without artefacts corrections has complexity:

$$O_{f,L} = O\left(\frac{L}{2}+1\right) + O\left(L \times log_2(L)\right), \tag{5.9}$$

where L is the length of the window. The first term accounts for the spectrum scaling operations, while the second term for the inverse Fourier transform.

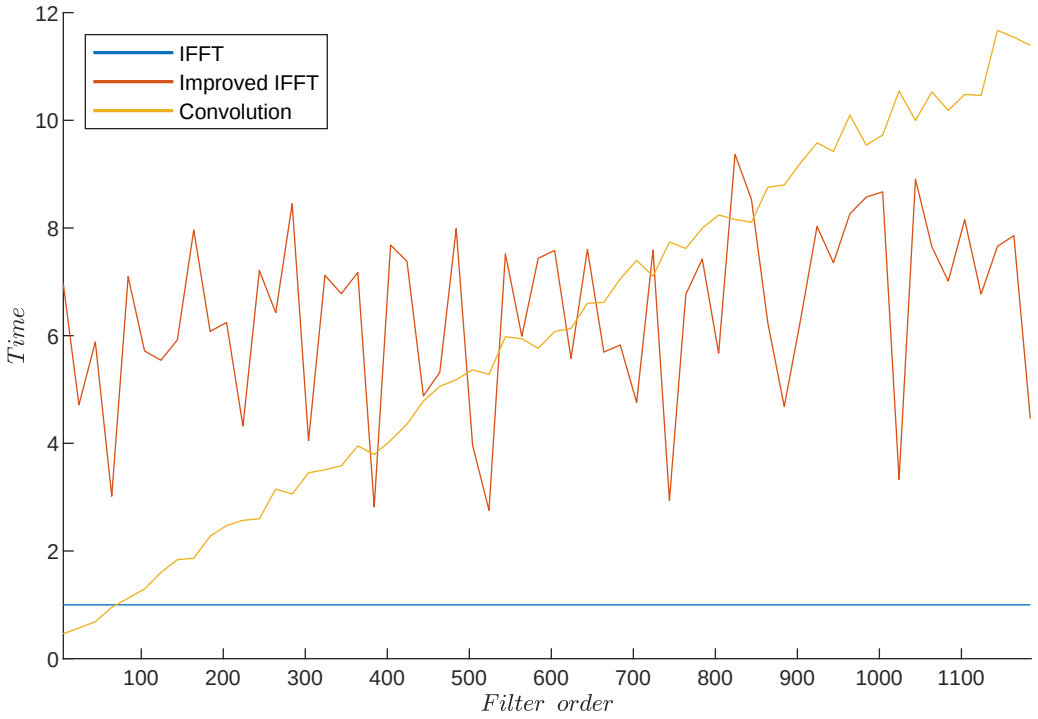


Figure 5.13: *Computational complexity. Filtering by inverse Fourier transform (blue line), by inverse Fourier transform with artefacts correction (red line), and by linear convolution (yellow line). Length of the window* $L = 4800$*. Time is normalised against the execution of the IFFT solution. Improved IFFT remains roughly constant for filters of low order but tends to increase for higher values. Strong irregularities are related to the different efficiency of the FFT algorithm for signal chunks of different lengths. Linear convolution is more efficient than IFFT for* F *below* $70 \div 80$ *and more efficient than the improved IFFT for* F *below* $500 \div 600$*. Values are averaged over* 5000 *repetitions.*

If the filter is improved including the correction of the artefacts, using a filter of order F, the equation above becomes:

$$O_{f,F,L} = O\Big((F+1)\times log_2(F+1)\Big) + O\Big((L+F)\times log_2(L+F)\Big) + O\Big(\frac{L+F}{2}+1\Big) + O\Big((L+F)\times log_2(L+F)\Big). \quad (5.10)$$

The first term accounts for the inverse FFT *(IFFT)* of the impulse response of the filter, the second for the FFT of the filter of length $L+F$, the third for the spectrum scaling, and the last for the anti-transform of the filtered signal. Equation 5.10 assumes that the FFT of the input signal ($L+F$ samples long) is already available. When F is small, the equation above is roughly equal to $2\times(L+F)log_2(L+F)$.

The analogous relation for filtering in the time domain is:

$$O_{t,F,L} = O\Big((F+1)\times log_2(F+1)\Big) + O\Big(L\times(F+1)\Big) \quad (5.11)$$

where the first term accounts for the calculation of the filter impulse response and the second term for the convolution.

Figure 5.13 compares the three filtering solutions above for $L = 4800$. Results are obtained using a demo application running in *Matlab* environment and repeating each step 5000 times. As expected, filtering using the linear convolution (yellow line) is slower than filtering in the frequency domain (blue line) for F above $70 \div 80$. However, if the correction of the artefacts is accounted for (red line), filtering by convolution remains more convenient for F below $500 \div 600$. Therefore, since a good performance can be obtained with filters of order $150 \div 300$, filtering in the time domain remains a convenient solution and is the one adopted for the software implementation. For instance, figure 5.8 reports the filter modelling with filters of order 11, 119, 299: the latter is very close to the shape of the full resolution filter. During the filter synthesis, the order of the filter is imposed by decimating the full-resolution filter mask: the whole spectrum is divided into $F+1$ intervals, leaving null and Nyquist frequencies in the non-decimated group of samples. Then,

the decimated filter is anti-transformed to obtain the impulse response and perform the linear convolution.

5.3.4.2 Filtering quality

Although it is not possible to provide a unique performance figure for any possible signal, a special case is analysed to evaluate the filter effectiveness. A bespoke software utility has been developed for this purpose, and the related block diagram is represented in figure 5.14.

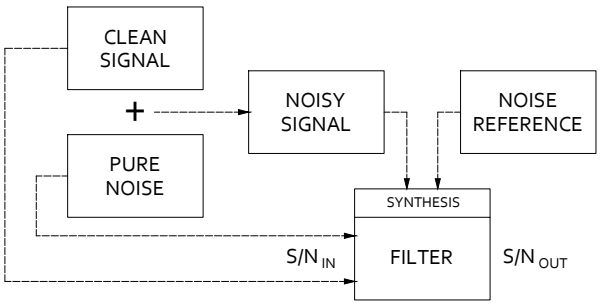


Figure 5.14: *Block diagram of the proposed software utility for the assessment of the filtering quality performance. A noisy signal is synthesised by combining a clean signal and pure noise. Then, the filter is synthesised from the noisy signal and the noise references. Finally, the clean signal and the pure noise are filtered separately using the synthesised filter.*

To obtain the real signal-to-noise ratio S/N before and after the filter, a $5s$ artificial *noisy signal* is synthesised by combining a *pure noise* from the soundbank and a *clean signal*. The clean signal is obtained by filtering white noise and retaining only the bands $[0.5kHz, 1.5kHz]$ and $[4.75kHz, 7.25kHz]$. The non-null clean signal is limited between 1 and 4 seconds by imposing a Chebyshev window to keep the frequency lobes down outside the frequency bands. The desired input signal-to-noise ratio is obtained by adjusting the amplitude of the clean signal in respect of the noise level. The pure noise signal selected is not included in the synthesis of the noise filter mask.

The S/N before the filter is simply calculated as the ratio of the clean signal energy to the pure noise energy. For the S/N after the filter, both the clean signal

and the pure noise are processed separately by applying the filter synthesised using the noisy signal (the *synthesis reference*) obtained as described above.

Signal S/N (dB)	Indicator (dB)	Filter	Test Id					Average (dB)
			1	2	3	4	5	
			SI 5 BG 0	TA 1 BG 0	TO 3 BG 0	SH 4 BG 0	WM 2 BG 0	
$0\ dB$	S/N gain	Liter. ref.	25.329	19.444	25.006	19.769	20.700	22.050
		Prop. equiv.	32.824	29.877	25.863	37.400	23.370	29.867
		Prop. default	36.053	30.232	30.878	40.340	23.729	32.246
	S att.	Liter. ref.	0.027	0.060	0.059	0.032	0.212	0.078
		Prop. equiv.	0.231	0.087	0.159	0.102	0.146	0.145
		Prop. default	0.430	0.155	0.316	0.176	0.237	0.263
	N att.	Liter. ref.	25.356	19.504	25.065	19.801	20.912	22.128
		Prop. equiv.	33.055	29.964	26.023	37.502	23.516	30.012
		Prop. default	36.484	30.388	31.194	40.516	23.967	32.510
$-20\ dB$	S/N gain	Liter. ref.	24.683	17.456	23.192	19.467	13.390	19.638
		Prop. equiv.	30.754	25.491	25.091	39.292	26.557	29.437
		Prop. default	37.915	32.515	32.936	41.988	27.256	34.522
	S att.	Liter. ref.	0.673	2.048	1.873	0.334	7.522	2.490
		Prop. equiv.	0.308	0.274	0.320	0.171	1.175	0.450
		Prop. default	0.503	0.414	0.547	0.241	2.035	0.748
	N att.	Liter. ref.	25.356	19.504	25.065	19.801	20.912	22.128
		Prop. equiv.	31.062	25.765	25.411	39.463	27.731	29.886
		Prop. default	38.418	32.930	33.483	42.229	29.290	35.270

Table 5.1: *Filter performance comparative table. The literature reference is compared against the proposed equivalent solution and the proposed default solution. Pure noise recordings are extracted from* 5 *different instances in the soundbank (`Sink 5`, `Tap 1`, `Toilet 3`, `Shower 4`, `Washing Machine 2`). For each instance selected, the first recording (`Background 0`) is used as pure noise, while the others are used for synthesising the noise filter mask. The clean signal is obtained by filtering white noise.* S/N *values are calculated using the clean signals and the pure noise before and after the filter. Results are calculated for* $0dB$ *and* $-20dB$ *input* S/N*. The attenuation of the noise and the attenuation of the clean signals are also indicated separately.*

As a consequence of the superimposition principle, the ratio of the energy of the filtered clean signal to the energy of filtered pure noise is exactly the value sought.

Table 5.1 compares the performance of the filter proposed against the literature reference [52] for five different noise examples extracted from the soundbank. Noise recordings are selected from five instances of five different classes. Per each instance, one recording is used as pure noise, while the others are employed to synthesise the noise filter mask. The configuration "*proposed equivalent*" is obtained by setting the filter parameters as close as possible to the "*literature reference*". For instance, the same excess factor ($E_{\%} = 1.5$) and the same Gaussian smoothing radius ($R_{FS} = 250Hz$ have been used). Filtering is obtained by single forward convolution. On the contrary, the "*proposed default*" reports the performance using the default configuration assumed in the software library. As shown, the S/N gain of the proposed solution always outperforms the literature reference for both the moderate noise case ($0dB$) and the strong noise case ($-20dB$).

$E_{T\%}$	$R_{FS}(Hz)$	Convolution	S/N gain (dB)	Signal attenuation (dB)
1%	150	Linear F/B	32.928	0.972
		Linear F	31.539	0.614
	250	Linear F/B	32.978	1.099
		Linear F	31.869	0.691
2%	150	Linear F/B	32.708	0.855
		Linear F	29.399	0.532
	250	Linear F/B	32.759	0.977
		Linear F	29.714	0.604
5%	150	Linear F/B	32.349	0.718
		Linear F	26.105	0.438
	250	Linear F/B	32.358	0.835
		Linear F	25.688	0.506

Table 5.2: *Filter performance as a function of some filter parameters. Other settings: windows length = 50ms, overlap = 50%, $F = 199$, $R_{FD} = 100Hz$, $R_{TD} = 200ms$, and $R_{TS} = 0.125ms$. Input signal $S/N = -20dB$.*

On the other hand, the literature reference exhibits slightly better behaviour for what concerns the attenuation of the clean signal in the $0dB$ case. This issue, however, is fairly negligible given the entity of the difference and the fact that the proposed solution attenuates less clean signal in the case of strong noise, when it is more important.

Table 5.2 reports the performance for different values of some of the filter parameters to understand how their variations affect the S/N gain and the clean signal attenuation. As reasonable, smoothing the density masks on larger frequency intervals makes the filter less precise, and part of the useful signal can be lost in the process. Similarly, reducing the $E_{T\%}$ limits the noise, but the higher threshold also affects the signal attenuation.

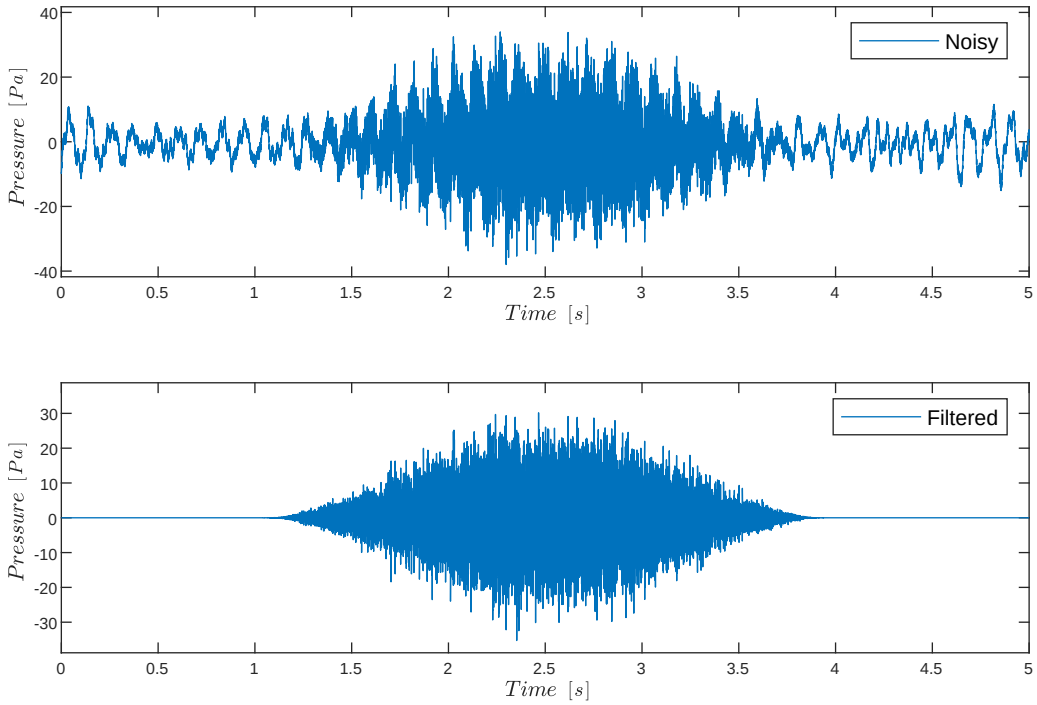


Figure 5.15: *Filtering example. Input $S/N = 0dB$, S/N gain $= 30.884dB$, clean signal attenuation = $0.326dB$. (Top) - Synthetic noisy signal. (Bottom) - Filtered signal.*

The convolution technique used can also have a strong impact and, although it comes with some drawbacks, performing a *forward/backward linear convolution*[4] generally introduces some relevant improvements on the S/N gain. The order of the filter should be chosen to provide enough resolution to follow the filter mask.

Resolution higher than necessary does not offer any significant improvement and increases the computational complexity. Table 5.2 is obtained using the default setting $F = 199$ for $L = 4800$.

Figure 5.15 illustrates a filtering example for a synthetic signal with $S/N = 0dB$. The filter settings are: $50ms$ windows, 50% overlap, $F = 199$, linear forward/backward convolution, $E_{T\%} = 1\%$, $R_{FD} = 100Hz$, $R_{TD} = 200ms$, $R_{FS} = 150Hz$ and $R_{TS} = 0.125ms$.

5.4 Synthesis of the observations

Machine learning systems need data to be trained and perform specific tasks successfully. When the dataset available is not sufficiently large, the probability of overfitting the training data and underperforming on an unknown test set is high. When a larger dataset is needed, a possible partial workaround could be extending what is already available by creating new artificial data. Although synthetic observations may not always be completely realistic, this method introduces some interesting benefits. Apart from extending the dataset size, *augmenting data* by combining and transforming elementary sound blocks allows accurate control of the resulting signals. For instance, labels can be assigned precisely, and acoustic properties, such as signal-to-noise ratio and event overlapping, can be decided beforehand. Besides, to obtain more realistic results, additional rules can be adopted during the synthesis, for example, by creating a kind of language model for the sound scene simulated [38, p. 161]. Nevertheless, the main drawback is generally the limited coverage of the soundbank used for the synthesis (see also section 6.3). Several examples of augmented datasets are available online [264],[249] along with the related software tools [58].

[4] To perform a *forward/backward* convolution, the filtering signal is convoluted once in the forward direction. Then the resulting signal is reversed, convoluted again and reversed back. Computational complexity is almost equivalent to a filter of double order, and the entire window must be acquired before filtering. However, apart from offering improved S/N, when the filter exhibits a complex frequency response, this convolution allows *zero-phase* filtering, meaning that filtering can be performed without phase distortions. Note that the filters proposed in this section are purely real and do not introduce phase distortion.

This section describes how the synthetic observations are generated in this work. A schematic representation is provided in figure 5.16, which illustrates how signals are processed along the chain. The `Sound Repository` is organised in three main archives: `Acoustic Models`, `Soundbank`, `Synthetic Datasets`. The first folder stores a set of results for the acoustic model obtained in chapter 4. The second folder stores the real recordings used as audio sources (see section 5.2). The last folder collects all the synthetic datasets created, with a new directory (`Synthetic Dataset n`) per each new dataset. Finally, every dataset instance is organised into two further sub-folders: `Synthetic Observations` and `Synthetic Soundbank`. The former stores the final results of the processing chain — the synthetic observations —, while the latter a copy of the synthetic audio blocks used for the synthesis (see section 5.2.4). The synthetic audio blocks are divided into *events*, *disturbances* and *backgrounds*, each having a different role in the output signal. The following subsections describe the techniques used to synthesise the observations and the structure of files and folders in the datasets. A description of the annotations is also provided.

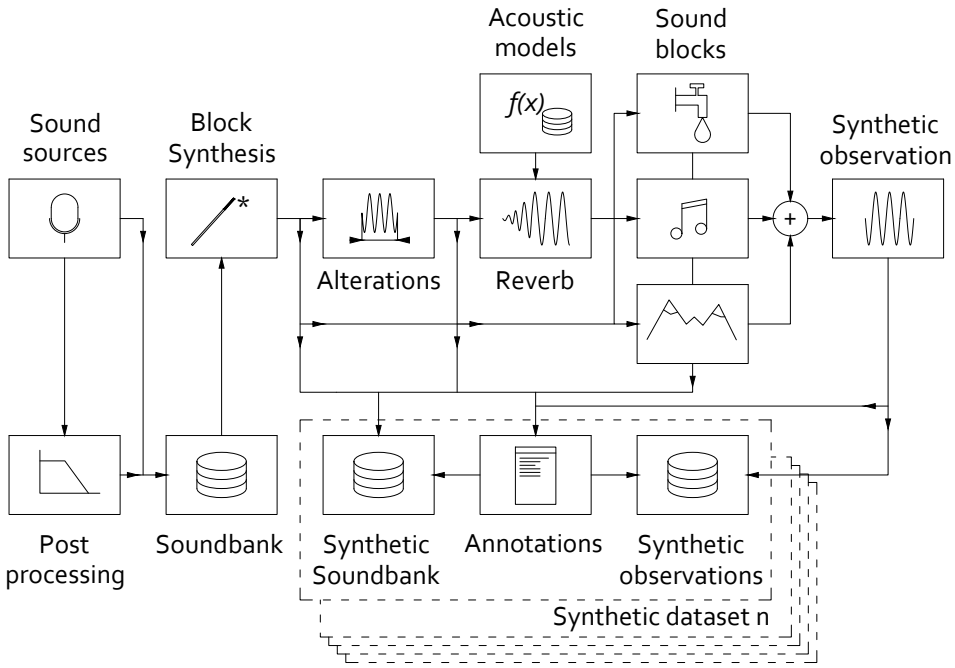


Figure 5.16: *Block diagram of the acoustic synthesiser.*

5.4.1 Semi-synthetic augmented observations

The recordings included in the soundbank can be recombined and transformed to obtain new semi-synthetic augmented observations organised in datasets arbitrarily large. A similar approach has already been employed in several contexts, but the way source signals are transformed and recombined together depends mainly on application constraints, such as the necessity to preserve harmonic features or to create meaningful time-structured events. A first possible solution is the one adopted by Salamon and Bello [265], which consists of pre-transforming the input audio signals by applying *pitch-invariant time stretching*, *time-invariant pitch-shifting* and *dynamic range compression*. Implementation details about these techniques are reported in [266] and [267]. The transformed signals are then superimposed to create the final synthetic observation. Another solution is the one employed by Parascandolo et al. [268], where transformations such as *sub-frame time shifting* and *block mixing* are applied on the extracted features in the frequency domain. A further example is the *Equalised Mixture Data Augmentation* [269] given by Takahashi et al., where new instances are obtained by mixing different events of the same class and by partial alteration of the spectrum.

The approach here used to generate synthetic observations for in-pipe events employs some of the solutions seen above but introduces elements of novelty dictated by the specific context. A *synthetic observation* is generated by combining a certain number of *synthetic event blocks*, a certain number of *synthetic disturbance blocks* and a *synthetic background*. The number of events and the number of disturbances per observation are obtained by randomly choosing them in the desired range. To synthesize an event block, a class instance is first selected from the soundbank. The new event is obtained by selecting a set of random `Event` items from the `Event Observations` folder, slicing them into parts, and combining a semi-random subset of slices together. When a certain class exhibits non-homogeneous or structured source observations, the random set includes only one randomly selected source observation (the option is regulated by the parameter `structured` in `Classes Attributes.txt`). This strategy guarantees more realistic results when classes include events with complex sound structures

that are unlikely to recombine properly. A typical sliced source event is represented in figure 5.17. To preserve the integrity of its structure, the event is sliced into three parts: *onset*, *regime* and *offset*. This novel approach is motivated by the need to preserve the transient peculiarities contained at the beginning and at the end of audio events.

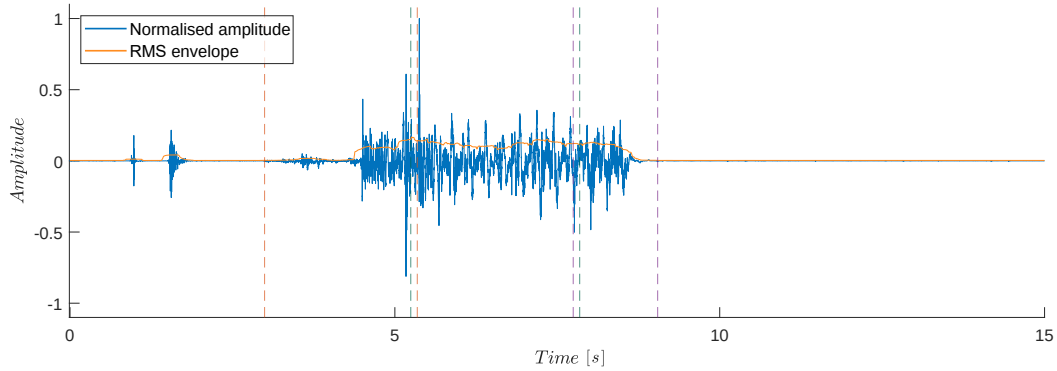


Figure 5.17: *A sliced event observation (normalised amplitude). The orange line reports the RMS envelope while the boundaries of the slices are marked with dashed lines of different colours (red:onset, green:regime, violet:offset). The main interesting part of the signal is isolated from other undesired sounds and the silent parts of the recording.*

Source signals are processed in windows of fixed length ($100ms$ by default), and the position of the splitters is determined semi-randomly using a mix of time, amplitude, and energy criteria. A minimum window energy threshold is first used to identify the beginning and the end of the main part of the signal and to isolate it from other undesired sounds or from the silent parts of the recordings. The main section of the signal is then extended to include adjacent sound bits in its temporal proximity. Once the boundaries have been identified, the other splitters can be marked. In particular, *onset* and *offset* are determined by the application of thresholds on the total energy, on the amplitude of the envelope, and on the average energy, along with other conditions on the minimum and on the maximum length for very long or very short recordings. In case of extra long events, a randomly positioned *regime* chunk is extracted between *onset* and *offset*. In the other cases, an *overlap region* of one window length is left between onset, regime and offset.

Once the signal chunks have been obtained, an onset and an offset are randomly

selected to set the beginning and the end of the new synthetic event. Several regime chunks are then stacked together to obtain a resulting event whose length is close to the desired one. For smoother joints, the best-joining windows are determined by comparing both spectrum and average amplitude (a larger weight accounts for the spectrum similarity). The amplitudes are then adjusted across the joining region to guarantee a smooth transition (figure 5.18). When only a single short event observation is selected as a signal source, the resulting synthetic event will be identical to the original one cleaned of the silent endings and other undesired sounds: this allows the inclusion of real observations among the synthetic ones. Once a new synthetic event has been obtained, its length is tweaked by pitch-invariant time stretching to match the randomly determined desired length. A random time-invariant pitch-shift is also applied to increase the variability of the results. The implementation of the time-stretch and the pitch-shift is the one given by the *Matlab* functions `stretchAudio` and `shiftPitch`, whose description is reported in [270]. In the last step, the artificial reverberation is applied as discussed in section 5.5.

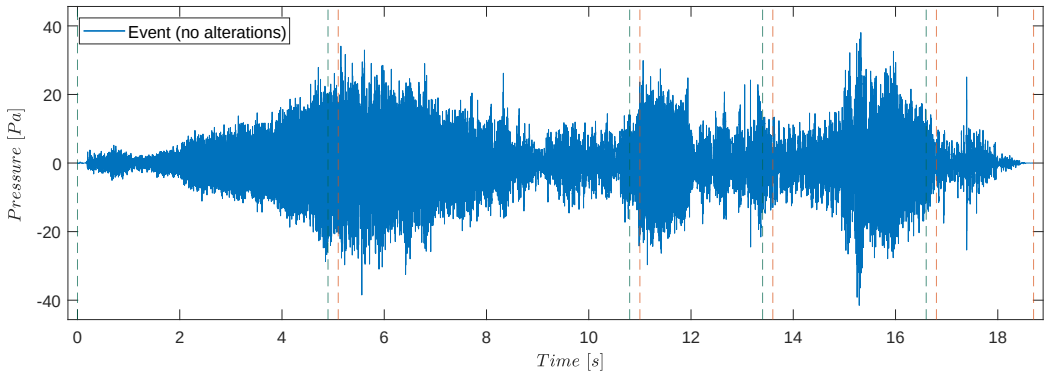


Figure 5.18: *Structure of a synthetic event. The endings are obtained from an onset chunk and an offset chunk, respectively. The central part of the signal is built by combining and matching adapted regime slices together.*

Once a synthetic event block has been generated, a set of audio files is stored for reference in the `Synthetic Soundbank` under the related `Synthetic Event` folder. The file `synthetic event (no alterations).wav` represents the synthetic event before time-stretch and frequency shift. The file `synthetic event (no reverberation).wav` represents the synthetic event before the reverberation (see

section 5.5). Finally, the file `synthetic event.wav` is the sound block used for the synthetic observation. Along with the audio files, a set of time-domain graphical representations of each step of the event synthesis is stored under the folder `Pictures` (figure 5.19). Moreover, an annotation file `annotations.jams` is also generated. Further details are reported in the following section.

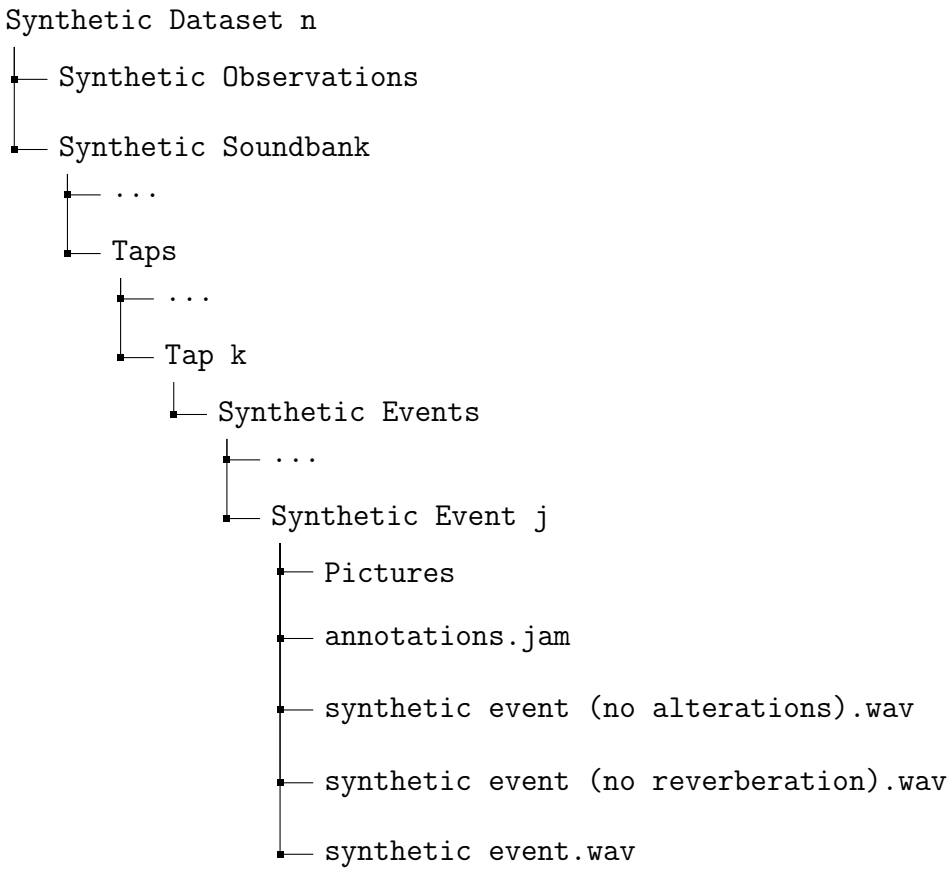


Figure 5.19: *Position of the synthetic events and related files in the synthetic soundbank of the dataset.*

Apart from the events, the other optional signals for synthesising a new observation are the disturbance blocks and a background block. The disturbances are randomly selected from the `Disturbance Observations` folder under the `Disturbances*` class of the `Soundbank` and processed to isolate the main part of the signal. Then, time-stretch and pitch-shift alterations are applied along with the reverberation (section 5.5). As for the synthetic events, each synthesis step is

saved as an audio file along with the related annotations and graphic time-domain representations. Each synthetic disturbance is stored in the `Synthetic Soundbank` in the folder `Synthetic Disturbances` under the class `Disturbances*`. Similarly, a source background is randomly selected from the folder `Background Observations` in `Backgrounds*` and replicated several times to reach the desired duration. No further transformation is applied in this case. The generated background is then stored in the folder `Synthetic Backgrounds` under the class `Backgrounds*`.

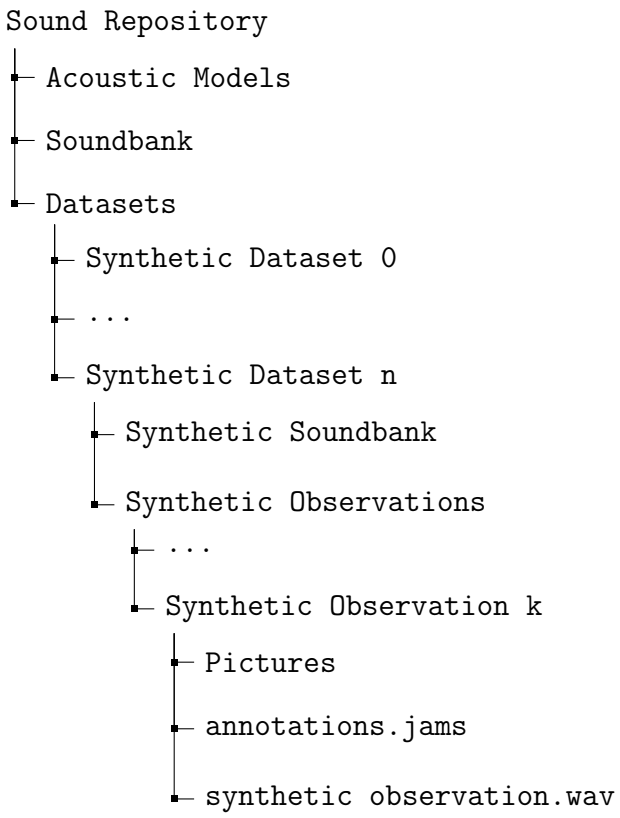


Figure 5.20: *Position of the synthetic observations and related files in dataset.*

Once all the required sound blocks have been generated, they can be superimposed to generate the synthetic observation. The synthetic background, whose length matches the duration of synthetic observation, is added first. Then, the position of events and disturbances is randomly determined. Knowing the position, it is possible to randomly assign (or to calculate) the scale factor to maintain the signal-to-noise ratio between the desired minimum and the desired maximum.

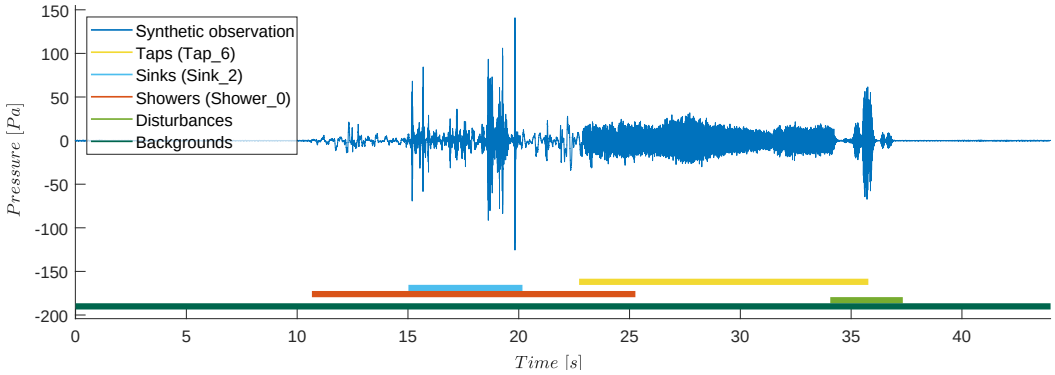


Figure 5.21: *Time domain representation of a synthetic observation. The timing of the sound blocks is marked by the lines underneath.*

Completed the task above, the resulting observation is placed under the `Synthetic Observations` folder of the `Sound Repository` (figure 5.20). The `Pictures` folder contains a time-domain representation of the synthesised observation with the indication of the position of the audio blocks (figure 5.21). The time-frequency representation is also generated as a spectrogram (figure 5.22).

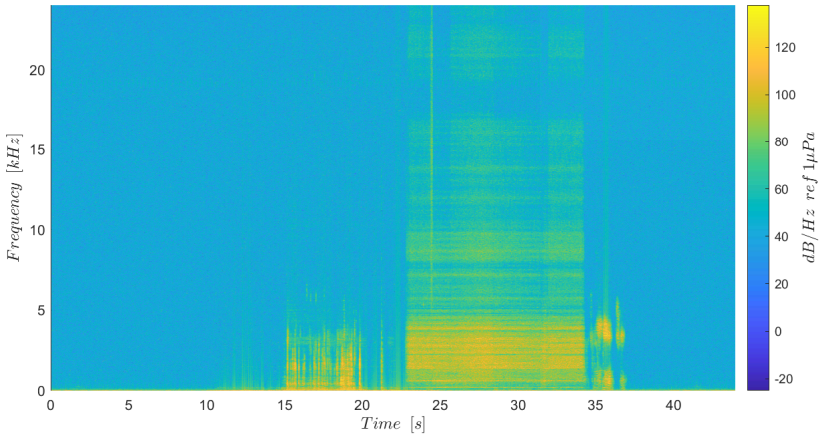


Figure 5.22: *Power spectral density spectrogram of a synthetic observation. Measures are reported in dB/Hz ref 1uPa.*

To maintain the coherence of the dataset, care must be taken when items in either the `Synthetic Soundbank` or the `Synthetic Observations` folders are modified. For instance, if a synthetic observation were removed, some synthetic events would remain in the `Synthetic Soundbank` without being included in any synthetic output. Vice-versa, if a sound block were removed, a synthetic observation would exhibit

missing references to the `Synthetic Soundbank`. Therefore, to allow modifications in these folders, the software library developed is equipped with software tools to remove the desired items while maintaining the content of both the `Synthetic Soundbank` and the `Synthetic Observations` coherent.

5.4.2 Annotations

One of the most common barriers to creating new datasets is producing strong, reliable annotations. In general, labels should guarantee [38, p.152]:

- *representation*, that is, a clear description of the classes in a given context,
- *non-ambiguity*, that is, a clear separation of instances of different classes.

When labels are assigned manually on existing recordings, apart from the human labour required, marking the exact position of the sound instances and distinguishing between different classes has a subjective component of uncertainty [38, p.157]. On the other hand, reliable and precise annotations are the key to training machine learning algorithms successfully. One more issue also concerns the way annotation data are structured and stored. In general, metadata should be easily machine and human-readable and should allow for the storage of complex structured information. Simple textual annotation can be accessed easily, but the lack of structure easily translates into increasing complexity. Alternatively, other formats proposed, such as XML [271] or RDF [272], may not be easily human readable.

To mitigate the issues mentioned above, apart from defining an unambiguous set of class names, the assignment of the labels is here executed automatically and without any human intervention. As seen in the previous section, the sound blocks are isolated and synthesised from a set of weakly annotated files containing only one class each. Therefore, during the synthesis, the timing and the class label of each sound instance are exactly known. Observations metadata are then stored using *Json Annotated Music Specification* (*JAMS*) [154],[273], a format originally created for music annotation but recently adopted also for several Scene Event Detection datasets [58],[274]. The *JAMS* format guarantees good flexibility to store structured information and, at the same time, human readability as good as for common *JSON* files. Annotations are created and stored according to the

version *JAMS-0.2* of the standard. Under this format, it is possible to extend the list of the possible *tasks* to be annotated by declaring a new *task namespace*, that is, a bespoke *partial JSON schema* that can be imported into the library and where the rules for the required *task annotations* are reported. In general, a task annotation is composed of a 4-tuple of fields: `time`, `duration`, `value`, `confidence`. The definition of a new namespace mainly concerns the rules for the field `value`, where bespoke data can be organised and stored.

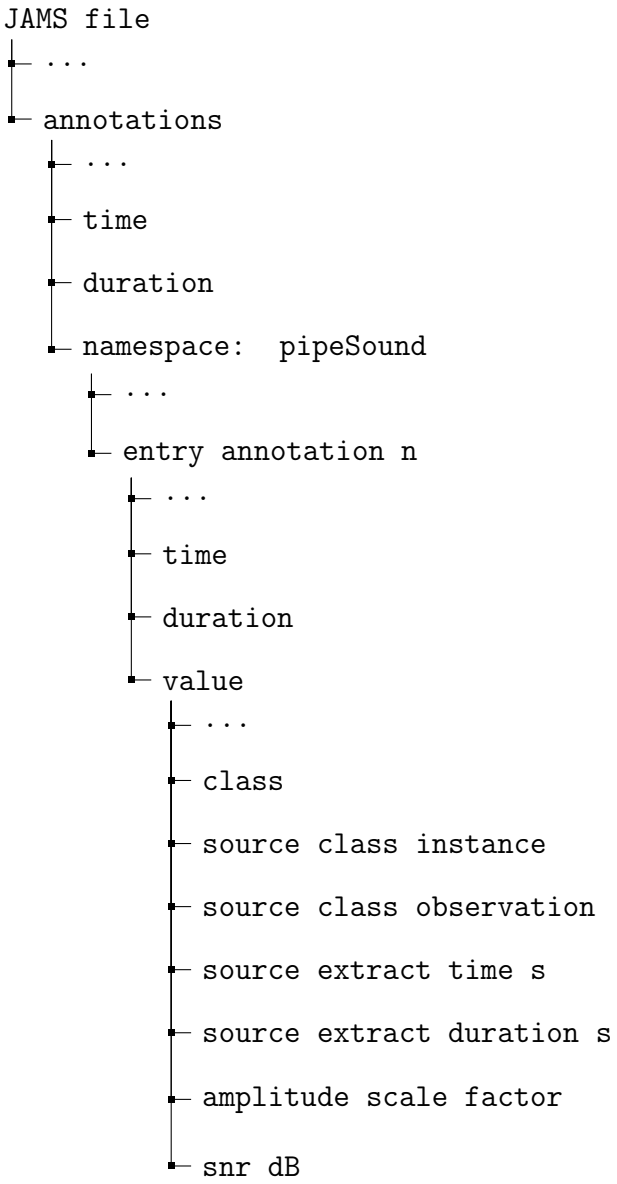


Figure 5.23: *Structure of the annotation JAMS file for a synthetic observation.*

Here, a new namespace named *"pipeSound"* is defined to keep track of the steps necessary to create the synthetic signals. The fields `time` and `duration` specify the beginning and duration of the sound block, while the field `confidence` is neglected. Each annotation entry in the field `value` describes a sound block composing the synthetic sound. The annotations of a synthetic observation include an annotation entry per each synthetic event, disturbance, and background composing the sound signal. Figure 5.23 highlights the main features of the structure of the `JAMS` file for a synthetic observation. The fields `class`, `source class instance` and `source class observation` identify the source of the specific sound block imported.

The fields `source extract time s` and `source extract duration s` identify the sub-part of the source file imported. Note that synthetic sound blocks are imported entirely into the synthetic observation, so the value `extract time` is zero and the `extract duration` matches the length of the sound block.

The `amplitude scale factor` stores the multiplication factor used to scale a sound block during the creation of the synthetic observation, while `snr dB` stores the signal-to-noise ratio of a sound block for the correspondent background.

Each sound block used in the synthesis is created along with the related `JAMS` file. Although similar, the annotations for sound blocks exhibit a few differences with respect to those for the synthetic observations described above. When synthetic events and synthetic disturbances are generated, each stacked chunk is annotated separately. Besides, further transformations are applied to the whole sound block to finalise the synthesis. These transformations are annotated as special entries that appear on top of the list. Figure 5.24 highlights the main features of a `JAMS` file for a synthetic event block. The fields `time stretch ratio` and `pitch shift semitones` of the first entry store the alteration factors for time-stretch and pitch-shift, respectively. Then, a new entry is added per each chunk stacked, and references to the source are annotated as for synthetic observations. In this case, however, `extract time` and `extract duration` are given by the actual position of the chunk in the source recording. Finally, details about the field `reverberation` are given in the next section.

A similar description can also be given for the annotations related to the synthetic

backgrounds. However, since backgrounds are synthesised by simply repeating the chosen signal up to the desired length, the annotation contains only a single entry.

All the JAMS files are generated using the official Python library [275] which has been wrapped into *Matlab* for the purpose.

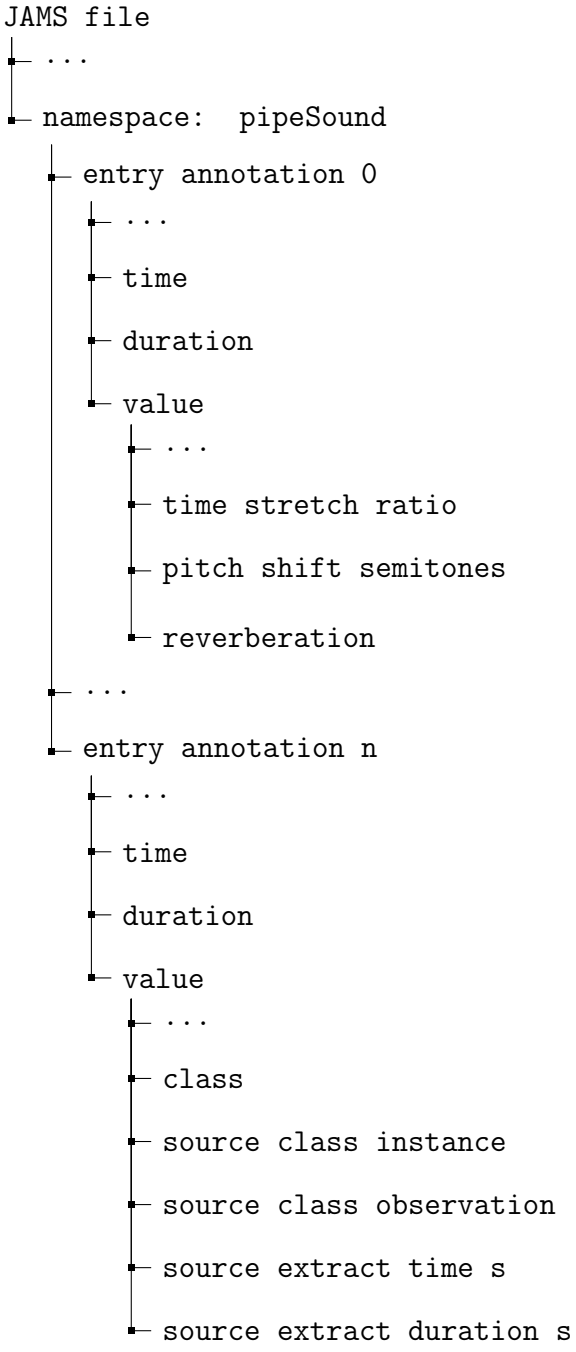


Figure 5.24: *Structure of the annotation JAMS file for a synthetic event.*

5.5 Adding reverberation by means of the acoustic model

As seen in chapters 2 and 4, sounds propagating in pipes are affected by the reverberation introduced by the presence of geometric boundaries. A similar scenario can be found in other applications where sounds propagate in closed or semi-closed spaces. Accounting for the reverberated sounds is an important requirement for applications such as improvement of room acoustics, video games audio rendering, production of datasets for indoor speech recognition, pipe status monitoring, and understanding of human sound perception in closed environments [59],[61],[276]. As shown in section 2.1.1, adding reverberation is generally performed using either a combination of calculation and real measurements or a fully simulated approach. In the former case, even not considering other practical acquisition issues [277], impulse responses depend on both the environment properties and the position of the source and the receiver [70]. Therefore, despite being more accurate, collecting impulse responses lacks flexibility when the aim is simulating a reverberated environment in different conditions.

In this work, a fully simulated approach has been preferred since, despite the limitations, considerable flexibility can be achieved without additional hardware costs or human labour. The same approach is generally preferred when the simulated conditions need to be changed in real-time or when there is a need to simulate the environment under different conditions or for different positions of source and receiver. For room reverberation, for instance, several solutions (generally based on image source models) have been proposed along with the related software libraries [59],[278]. Note that perfect simulations of real environments might be unfeasible, even with the aid of simulators using fully numerical methods. However, depending on the application, a simplified representation might be enough to account for the propagative distortions. Besides, if the simulated setup is simple, it can be replicated in a real test rig to verify the results.

As seen in chapter 4, the approach used in this work aims to simulate the reverberation in the simple case of straight elastic pipes where a circular piston represents the pressure source distribution. This solution provides a fair represen-

tation of the in-pipe reverberations and allows for matched measurements from real test rigs. Moreover, although this point has been left for future developments, real measurements can be integrated to improve the accuracy, for example, by acquiring the measured amplitude of the frequency response of the source. When solving the model, numerical calculations are performed only once and beforehand. The results are stored under the folder `Acoustic Models` (figure 5.4) and retrieved each time a new synthetic observation is to be generated. One simulation archive is saved per each simulation setup, that is, per each configuration of materials, geometries and radial position of the source. Numeric results relate mainly to mode amplitudes and wavenumbers. Results are calculated in frequency steps up to the chosen frequency limit providing frequency resolution chosen as a trade-off between accuracy and calculation speed. During the synthesis of the observations, only the third layer of the model is used. This makes the time required for the simulated reverberation shorter than the time necessary for the other tasks of the synthesis.

5.5.1 Simulated impulse response

Recalling section 4.4, the approximated output pressure in a point G for an input signal $x[i_t]$ can be written as:

$$\hat{\hat{p}}[i_f] = \hat{x}[i_f]\hat{\hat{p}}_x[i_f] = \tag{5.12}$$
$$\hat{x}[i_f]S[i_f]\rho^2 c_\phi (2\pi f[i_f])^2 \sum_{n=-N}^{+N}\sum_{m=1}^{+M_n} \bar{\Phi}_{nm}[i_f]J_n(q_{\phi_{nm}}[i_f]r_G)e^{in\theta_G}e^{ik_{z_{nm}}[i_f]z_G}.$$

In the previous equation, r_G, θ_G and z_G are the continuous space coordinates of the output pressure point G, while $[i_f]$ is the index of the discrete frequency vector $f[i_f]$. The transfer function $S[i_f]$ is defined according to the input signal $x[i_t]$. Here, $x[i_t]$ is a sequence of pressure values corresponding to a "dry" signal captured as described in section 5.2. To define $S[i_f]$, it is first defined the function:

$$A[i_f] = max\Big(\rho^2 c_\phi (2\pi f[i_f])^2 \sum_{n=-N}^{+N} \sum_{m=1}^{+M_n} \bar{\Phi}_{nm}[i_f] J_n(q_{\phi_{nm}}[i_f] r_G) e^{in\theta_G}\Big)[i_f], \quad (5.13)$$

which corresponds to the maximum pressure on the radiating surface scaled by its velocity $\hat{v}[i_f]$. $S[i_f]$ is assigned according to the three following criteria:

Model normalised: $S[i_f]$ is constant and assigned as:

$$S[i_f] = \frac{1}{max(A[i_f])}, \quad (5.14)$$

which means that the velocity $\hat{v}[i_f]$ is chosen so that the input pressure — at the frequency where the speaker pressure is maximum — matches the maximum pressure on the speaker. Using this definition, the simulated impulse response of the speaker is not altered.

Unitary: $S[i_f]$ is chosen as function of the frequency as:

$$S[i_f] = \frac{1}{A[i_f]}. \quad (5.15)$$

This option alters the model results but assumes that, per each frequency, the maximum pressure on the speaker surface matches the pressure of the input signal. This is the option used by default for the synthesis.

Measured: $S[i_f]$ is chosen as:

$$S[i_f] = \frac{A_m[i_f]}{A[i_f]}, \quad (5.16)$$

where $A_m[i_f]$ is the dimensionless measured amplitude, which describes the frequency response of the speaker. This option alters the simulated results but provides an impulse response more similar to the one associated with the real pressure source. Since $A_m[i_f]$ should be determined experimentally, this option is included in the software library, but it is not further investigated.

Once the transfer function has been defined, equation (5.12) can be used to obtain the pressure in the time domain. To calculate the product above, $\hat{x}[i_f]$ and $\hat{\tilde{p}}_x[i_f]$ must have the same number of samples and be defined in the same frequency range. Since $\hat{x}[i_f]$ is the input signal, the number of samples is determined by the related duration and sample rate. Therefore, $\hat{\tilde{p}}_x[i_f]$ is resampled by adapting f, $\bar{\Phi}_{nm}$, $q_{\phi nm}$, and $k_{z_{nm}}$. To retrieve the reverberated signal in the time domain, it is sufficient to perform the inverse fast Fourier transform and select the real part. When calculating the output pressure away from the source, the signal energy associated with the evanescent modes does not reach the receiver. Indeed, a small distance is sufficient to account for the exponential dumping.

5.5.2 Obtaining reverberated signals

When $x[i_t]$ is reasonably short, the whole signal can be processed at once (*single window*). To limit the artefacts introduced by truncation and circular convolution and to allow for the reverberation time, the input signal can be padded at its endings.

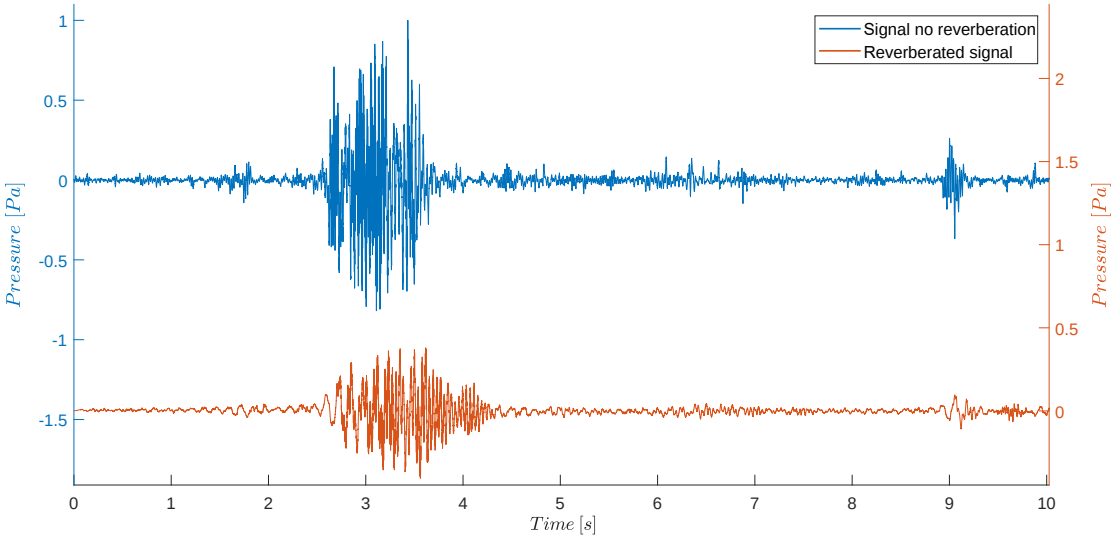


Figure 5.25: *Reverberation of a synthetic sound block. Aluminium pipe filled with water, outer pipe diameter* 0.1016*m, pipe thickness* 0.0016*m. Source transfer function: source model normalised. Reverberation distance* 500*m. Speaker and receiver in axisymmetric position.*

Figure 5.25 shows an example of single window processing for a signal of a few seconds. Since no attenuation is considered, the reverberation distance has been chosen long enough to highlight the delay and the tail in the reverberated signal.

If the signal is too long or other processing requirements arise (e.g. variable distance between source and receiver), the reverberation can be processed by dividing the signal into smaller windows. The technique proposed below is an adaptation of the *overlap-add* solution described for the noise filter (section 5.3.3) and commonly adopted in the literature for STFT signal reconstruction [279],[280].

In typical signal processing involving the STFT, the input signal $x[i_t]$ is first sliced into smaller overlapping chunks of fixed length. Then, a windowing function is applied to reduce the frequency-domain artefacts that would be associated with a squared truncation. The aim is to process each window in the frequency domain, and rebuild the signal back in the time domain (section 5.3.3). When a signal reverberates, however, a certain time shift is introduced, and the reverberation tail extends the duration of the output signal towards its end (section 4.4). When the operator $\hat{\hat{p}}_x[i_f]$ multiplies the windowed signal $\hat{x}_h[i_f]$ in the frequency domain, the circular convolution is calculated.

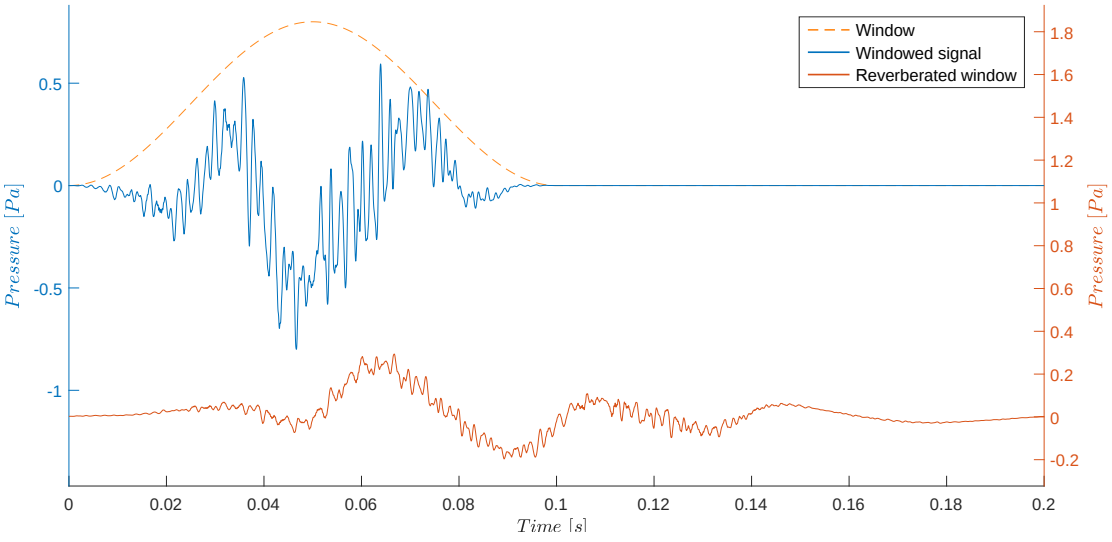


Figure 5.26: *Multi-window calculation of the reverberated signal. The interval of the extended window is long enough to cover the effects of the reverberation. Signal window length* 100*ms, extended window length* 200*ms, window function Hann window.*

Therefore, applying the reverberation directly on the windowed input signal would produce an undesired wrap-around effect in the time domain and, consequently, invalidate the results. To overcome this issue, the window ϕ of length L_ϕ (*the basic window*) is extended with zeros (padded) for an integer number k of extra lengths L_ϕ. The number k is chosen to allow for the time necessary to extinguish the reverberated signal. Note that, because of the propagation, the reverberated signal extinguishes even if no dumping is accounted for.

Figure 5.26 (top) illustrates the *multi window* processing where a Hann window $\phi[i_t]$ of $100ms$ is extended for extra $100ms$. The bottom signal shows how the reverberation causes the extension in the null area beyond the original window and how the extended window $\overline{\phi}[i_t]$ avoids the wrap-around. The frame h of extended windowed signal in the time domain can be written as:

$$\bar{x}_h[i_t] = x[i_t + hH]\overline{\phi}[i_t], \tag{5.17}$$

where h indicates the index of the extended windowed frame, and H is the hop. As shown in section 5.3.3, if the chosen extended window and the related hop satisfy the COLA condition ($\sum_h \overline{\phi}_h[i_t] = \sum_h \overline{\phi}[i_t - hH] = 1$), then the reverberated signal can be rebuilt as the sum of the reverberated windows:

$$\tilde{p}[i_t] = \sum_{h=0}^{F-1} \mathscr{F}^{-1}\left(\hat{\bar{x}}_h[i_f]\hat{\bar{p}}_x[i_f]\right)[i_t - hH]. \tag{5.18}$$

For the extended window $\overline{\phi}[i_t]$ described above, the COLA condition requires choosing the basic window $\phi[i_t]$ and the related hop to satisfy the COLA, and the hop of the extended window equal to the hop of the basic window.

5.5.3 Reverberation annotations

All the relevant reverberation properties used for the simulation are annotated under the `reverberation` field of the `pipeSound` namespace (figure: 5.27). The fields `pipe material`, `inner liquid`, `rho liquid` (kg/m^3), `c liquid` (m/s), `rho pipe` (kg/m^3), `cl pipe` (m/s), `cs pipe` (m/s) describe the physical properties of

the pipe and of the inner liquid. The fields `pipe outer diameter` (m) and `pipe thickness` (m) report the size of the pipe. The field `source spectrum amplitude` annotates the source normalisation method as described in section 5.5.1.

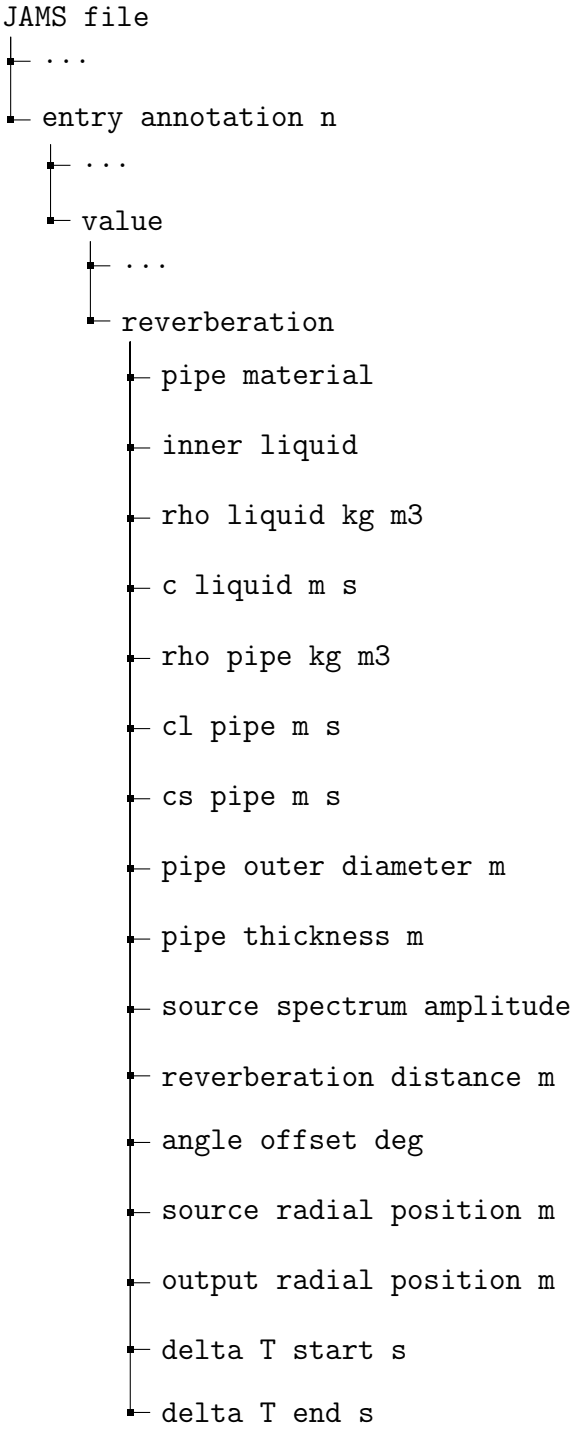


Figure 5.27: *Reverberation annotations in JAMS files.*

The fields `reverberation distance` (m) and `angle offset` (deg) give the relative distance between source and output point, while `source radial position` (m) and `output radial position` (m) the related radial positions. Finally, the fields `delta T start` (s) and `delta T end` (s) store the delay and the tail extra time introduced by the reverberated signal with respect to the original one.

The *delta T* fields are used to improve the accuracy of the annotation in the final synthetic observation so that the time shift and the distortion introduced by the reverberation do not affect the timing accuracy of the annotation. Unfortunately, time shift and tail also depend on the frequency components of the input signal and, regardless of the other simulation parameters, they cannot be assumed fixed. To overcome this issue, the initial time shift is found as:

$$\Delta T_{start} = T(max(corr(x[i_t], y[i_t])) - T(max(corr(x[i_t], x[i_t]))) \tag{5.19}$$

that is, by assessing the time difference between the maximum of the cross-correlation between the input and the output signal and the maximum of the auto-correlation of the input signal. On the other hand, the extra time introduced by the tail is evaluated by comparing the time necessary to reach the 90% of the signal energy for both the input and the output signal.

Note that the reverberation is randomly added only to synthetic events and synthetic disturbances, and the reverberation annotation appears only in the related `JAMS` files. In these annotations, the duration of events or disturbances differs from the related source audio blocks by a time given by *delta T end*. Then, *delta T start* is used during the synthesis of the observation to mark the exact beginning of the audio instance.

5.5.4 Limitations and benefits of the simulated approach

Adding reverberations to the signals of the dataset is undoubtedly helpful in accounting for in-pipes propagative effects. However, although the implemented model embeds some advanced features, it remains an extreme simplification of what can be found in reality. Although the acoustic mechanism underlying the

reverberation is the same, the final audio signal is affected by other factors, such as real geometries, real materials, obstacles, temperature, viscosity, and many others. For instance, the effects of attenuation have been completely neglected, and this issue certainly affects the modal components of the received signals [89]. Obviously, it is possible to improve the model, for example, by accounting for sound scattering from obstacles or by introducing more realistic sound sources as described in section 2.1.1. On the other hand, however, additional complications would increase the computational costs and are also likely to make matching with a real test rig more difficult. An interesting point to investigate might be understanding how a more accurate model would improve the accuracy of the machine learning algorithms trained using the dataset obtained. Finally, a heuristic approach (e.g. recording impulse responses) certainly provides a better representation of the sound scene but reduces the flexibility and increases the requirements for hardware and human labour.

5.6 Summary

This chapter describes the proposed methodology for creating an arbitrary number of semi-random augmented synthetic observations starting from a relatively small collection of source recordings. Firstly, the procedure followed for the acquisition of the soundbank is explained. The acquisition chain is analysed to clarify the motivations behind the hardware devices employed. It is shown how the human tasks required to gather the metadata are minimised and reduced to the storage of audio files in the related class/instance folders. A statistical spectral filter to attenuate the background noise is proposed to enhance the quality of the audio blocks and improve the accuracy of the audio synthesis. Two novel algorithms are described for the creation of noise masks and time/frequency density masks. The pros and cons of filtering in the time and frequency domain are analysed along with the conditions required for proper signal reconstruction. A bespoke methodology is proposed to measure the filter performance regarding computational complexity and filtering quality. For the latter, the signal-to-noise ratio for both moderate and high noise levels indicates a clear improvement of several decibels against the

closest example in the literature. The diagram of the novel audio synthesiser is illustrated and discussed. The recombination and the synthesis of the audio blocks, and the strategy adopted to control properties such as signal-to-noise ratio, duration, and the number of overlapping events are explained. The acoustic model developed in chapter 2 and 4 is integrated to simulate in-pipe reverberation. A solution to perform the related calculations in short windows is proposed. Finally, the annotations, the proposed extension of the `.JAMS` format, and the strategy adopted to keep track of the synthesised source blocks are described. The limitations of the approach adopted are also briefly discussed.

Chapter 6

Dataset and event classification

In the previous sections, it has been shown how to turn a collection of sounds into hard-labelled synthetic augmented observations. The reverberation effects have also been included to simulate sound propagation in an in-pipe environment. The last part of this work aims to analyse the recordings to automatically classify the acoustic events by means of some machine learning techniques. The concepts introduced in chapter 3 are here recalled, applied to the new datasets, and extended to improve the performance. In the first part of the chapter, data are synthesised, extracted, and organised for machine learning tasks. Then, the adopted processing chain is introduced along with the signal representations and the reference classifiers employed. The performances of the feature solutions developed are compared and reported as a benchmark for future developments.

6.1 Dataset synthesis and partitions

When machine learning techniques are tested against a specific dataset, it is usually necessary to partition the recordings in at least a *training set* and a *test set*. The former is used to train the algorithms that perform a specific task, while the latter is used to assess the performance. To obtain a reliable evaluation, it is important to keep the test and the training set separate. Indeed, if some recordings are shared between the test and training sets, overfitting can be easily confused

with the ability to generalise to unknown observations. Besides, as shown in chapter 3, it is desirable to perform training and test operations on datasets where the different classes are sufficiently represented and balanced.

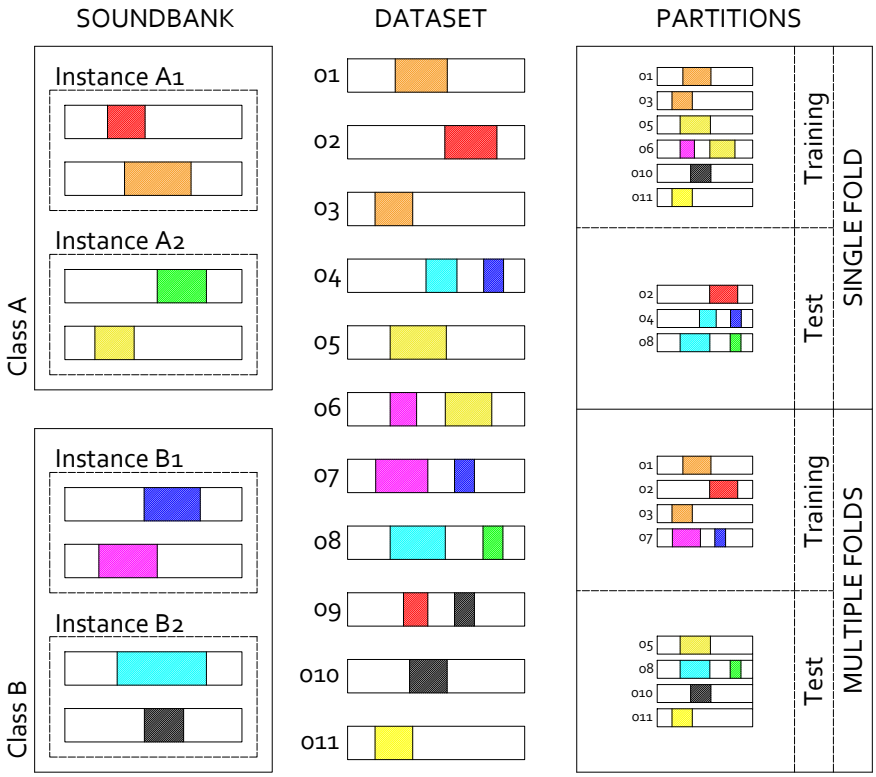


Figure 6.1: *Dataset synthesis and partitioning. The single-instance foldable synthesis guarantees that instances are evenly distributed in the synthesised dataset. Single-fold partitions are obtained by dividing the source observations in the soundbank in training (orange, yellow, violet, black) and test (red, green, blue, azure) observations. Multiple folds partitions are obtained by dividing the source instances in training ($A1$, $B1$) and test ($A2$, $B2$) instances. In different folds, instances are included (in turn) either for testing or for training purposes. Synthetic observations where test and training source blocks are mixed up are not included in the partitions.*

To guarantee the properties above for a synthetic dataset, further considerations are required. Generally, a class is assumed to be adequately represented when a sufficiently large number of examples are given. For a synthetic dataset, however, although the number of examples can be arbitrarily expanded, the related ability to represent a specific class is, to some extent, limited by the number of underlying source observations in the soundbank. Besides, defining the training set and the test set by simply dividing the synthetic dataset into two non-overlapping groups

of synthetic observations might lead to unreliable overfitted performances. For example, two different synthetic observations (e.g. `Synthetic Observation 0` and `Synthetic Observation 1`) can share the same source instances (e.g. `Tap 1`) and even part of the same source observations (e.g. `Event 3`). Therefore, to operate a useful partition of the synthetic dataset, it is necessary to understand how to manage the partition of the underlying source blocks.

6.1.1 Single instance foldable synthesis

To be effective, the solutions proposed for the partition of the dataset rely on a synthesis strategy here called *single-instance foldable synthesis* (figure 6.1).

To optimise the usage of the data available, a common method adopted consists of creating different *folds* of the same dataset (*cross-validation*) [281],[282]. In different folds, the same observation can be used, in turn, either for training or testing purposes. If properly implemented, this allows training and testing over the whole dataset while limiting possible overfitting issues. Although this solution can be helpful in the case of small datasets, it is important to understand how partitions are defined to reliably characterise the performance [283],[284]. From this point of view, tuning a synthetic dataset can facilitate the partitioning process. Indeed, instances that are under-represented in the soundbank can be rebalanced to be adequately separated at the instance or at the event level. A balanced representation is also beneficial in the case of single-fold partitions since the separation can be directly obtained without specific concerns about the size of particular instances.

With the *single-instance foldable synthesis*, the dataset is synthesised while ensuring that each instance of each class is evenly represented in the dataset (figure 6.1). For single-class synthetic observations, this strategy allows the creation of non-overlapping partitions obtainable even from randomly picked class instances. Note that, since the soundbank is usually unbalanced, synthesising the observations as described above generates skewed datasets, meaning that some classes will be associated with more observations than others. This issue, however, can be fixed as shown in the next sections. Table 6.1 reports the composition of the

soundbank used in this work. Figure 6.2 illustrates the summary of a single-class (plus background) dataset generated by single-instance foldable synthesis from the soundbank of table 6.1.

Class \ Instance	0	1	2	3	4	5	6	7	8	9	10
Backgrounds	459										
Disturbances	84										
Dish Washers	2	2	2	2	-	-	-	-	-	-	-
Showers	2	11	13	17	11	10	15	-	-	-	-
Sinks	6	7	3	2	2	9	10	12	10	11	10
Taps	12	12	9	9	8	9	10	9	9	11	10
Toilets	5	8	3	10	10	10	12	-	-	-	-
Washing Machines	40	17	18	47	18	23	-	-	-	-	-

Table 6.1: *Composition of the soundbank. Some classes are better represented than others making the source archive unbalanced. Backgrounds and disturbances are acquired along with the other classes but joined all together as unique instances.*

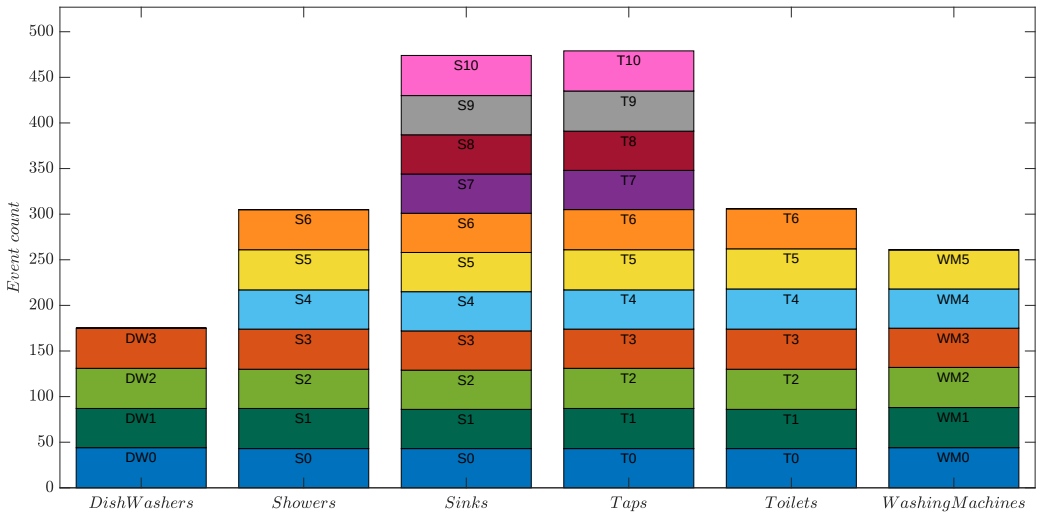


Figure 6.2: *Summary of a dataset of* 2000 *single-class synthetic observations generated with single-instance foldable synthesis. No disturbance is included. The dataset imbalance reflects the imbalance of the soundbank since each instance is evenly represented in the dataset.*

6.1.2 Dataset partitioning

Once the dataset has been generated, it is possible to create the partitions by analysing the source blocks associated with each observation. In this work, partitions are created in single or multiple folds as described below.

The *single-fold* approach (figure 6.1) is similar to a simple *test/training split* [283]. In this case, however, the dataset is not partitioned by simply dividing the synthetic observations into test and train sets. Indeed, this solution would potentially gather synthetic observations whose source blocks share part of the same source audio recordings between both the training and test set. Instead, the synthetic observations are first analysed to check the source items included. Then, the included source recordings (events and disturbances) are divided into the desired proportion to represent each instance in both the training and the test set. Finally, the synthetic observations are assigned to either the test or the training set to achieve disjoint source items. Note that synthetic observations including source blocks obtained from source events or disturbances belonging to both the training and the test sets will be excluded from the partitions.

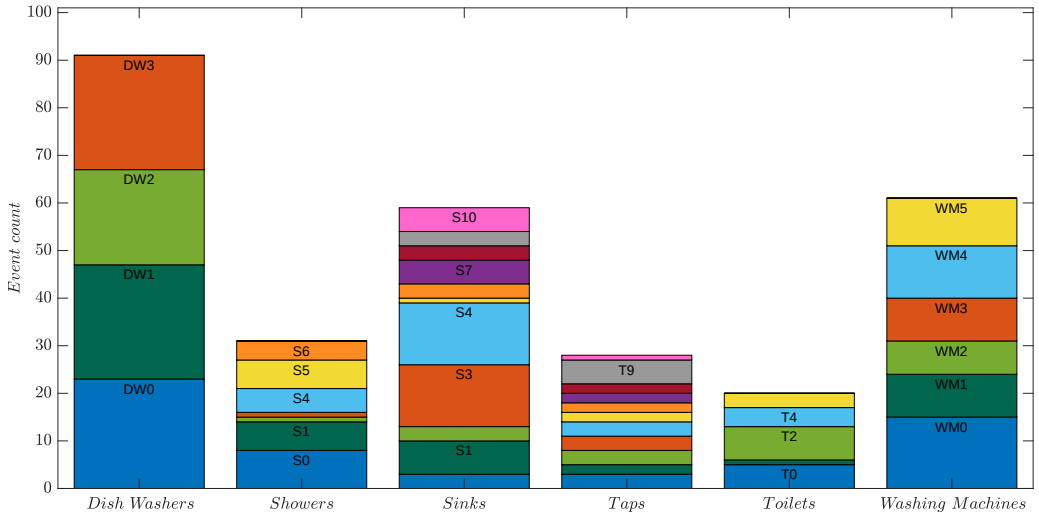


Figure 6.3: *Unbalanced single-fold test partition for the dataset reported in figure 6.2. Desired percentage test split = 25%. The number of synthetic events included also depends on the number of source events per instance in the soundbank and on the number of synthetic observations excluded due to the inclusion of both test and training source recordings.*

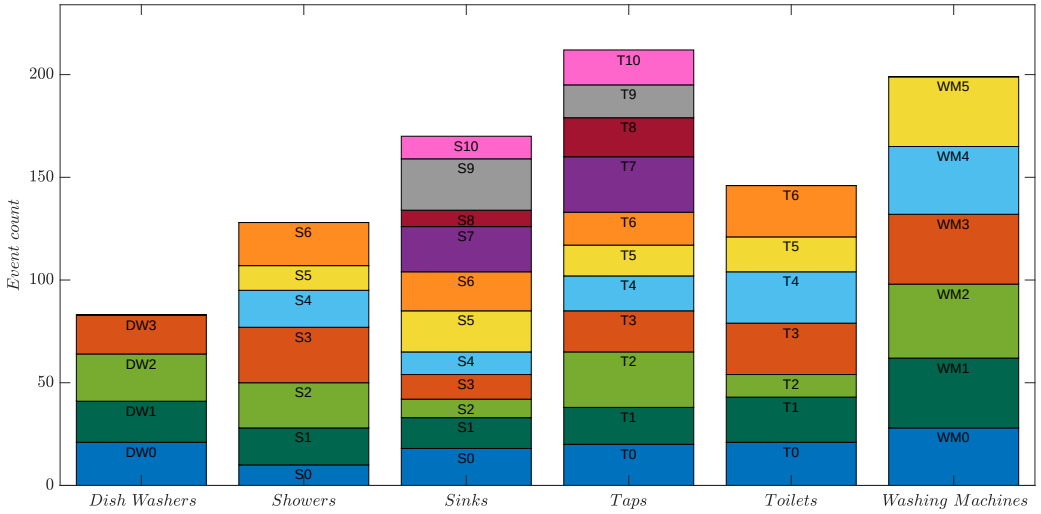


Figure 6.4: *Unbalanced single-fold training partition for the dataset reported in figure 6.2. Desired percentage training split = 75%. All the instances are represented in the training set.*

Since the dataset is unbalanced, the generated partitions will also be unbalanced. The balance, however, can be reestablished by randomly picking the same number of observations per class. Compared to a traditional test/training split, this solution guarantees disjoint source audio items and the presence of all the instances in the two partitions. Note that, although having the same instances in both the train and test set increases the representation of the classes in the partitions, this solution might penalise the reliability of the performance since source blocks belonging to the same sound source (e.g. `Tap 0`) can be used for both training and testing purposes. Figures 6.3 and 6.4 represent an example of unbalanced partitions for the dataset of figure 6.2.

The *multiple-folds* approach (figure 6.1) is similar to *nested cross-validation* [283], which generally provides better reliability over other forms of multi-fold cross-validation [285]. However, similarly to the single-fold case, the partitions are obtained by randomly separating, in a chosen proportion, the *source instances* in training and test source instances. The number of folds is chosen to make the test partitions cover the whole dataset. For instance, if the desired test proportion is 25%, the corresponding number of folds will be 4. Depending on the number of folds, some instances might overlap in the first and in the last folds (remaining

always disjoint between the test and training set). By considering the partitions of all the folds together, it is possible to account for all the instances in the dataset while always having disjoint audio sources in the test and the training set. Similarly to the previous case, multi-class (or multi-instance) synthetic observations that include both training and test instances are not included in the partitions.

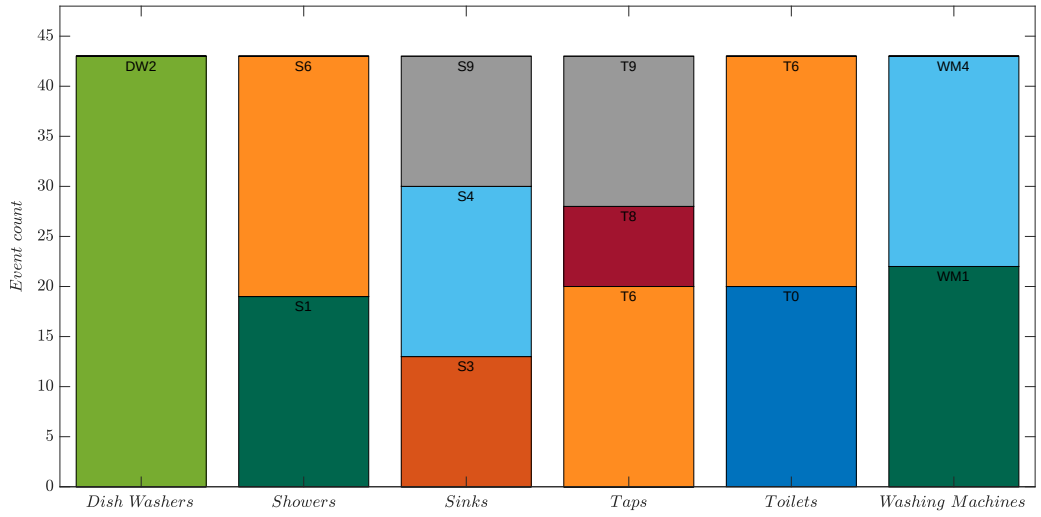


Figure 6.5: *Balanced 4-folds test partition for the dataset reported in figure 6.2. Fold 1/4. Test desired percentage split = 25%. The instances included in the test set are not included in the train set and vice-versa, that is, the test and the training set include disjoint audio sources.*

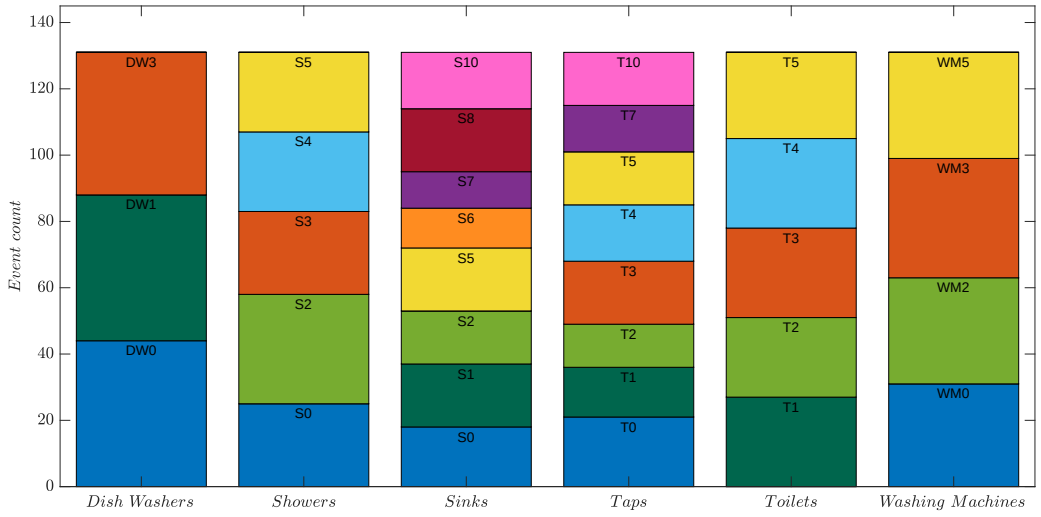


Figure 6.6: *Balanced 4-folds training partition for the dataset reported in figure 6.2. Fold 1/4. Training desired percentage split = 75%.*

Therefore, the set of observations included can be different between two different folds. No observation is discarded when the dataset is synthesised with a single instance per observation. Note that, when disturbances are accounted for in the creation of the partitions, since no instance is defined for them, they are divided by the related source observations as in the single-fold case. Finally, note that the created partitions are generally imbalanced, but the balance can be reestablished by randomly selecting the same number of synthetic observations per class. Figures 6.5 and 6.6 show the balanced multi-fold partitions for the fold $1/4$ of the dataset reported in figure 6.2.

6.2 Single class discrimination

In this section, the concepts introduced in chapter 3 are recalled and applied to understand how different signal representations perform with respect to classification tasks related to the previously generated data.

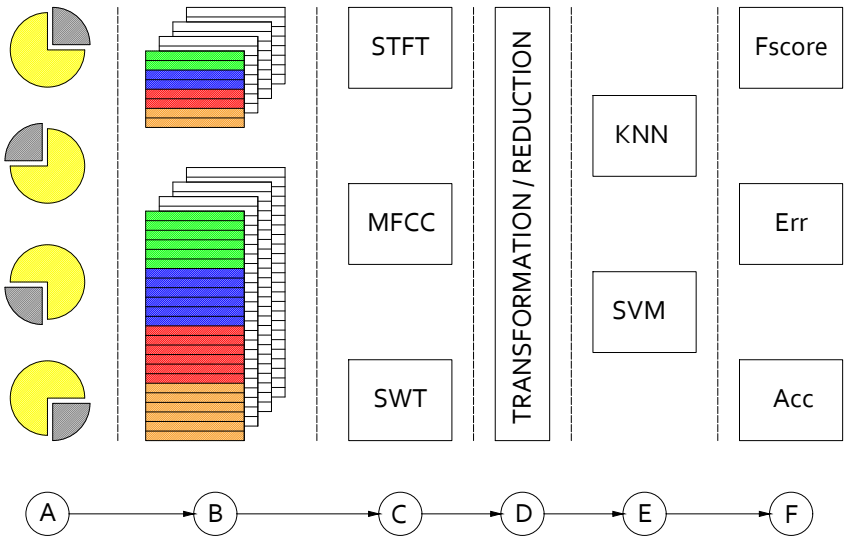


Figure 6.7: *Processing chain for single class discrimination. First, the partitioned synthetic observations (A) are organised in balanced collections of examples (B). Then, (C), the time domain signals are represented using one among STFT, MFCC and WST. The obtained representations are transformed/reduced in a set of features (D) which are employed to feed either the KNN or the SVM classifier (E). Finally, step (F), the performance is assessed using F-score, error rate and accuracy.*

Figure 6.7 provides a schematic representation of the processing chain adopted. First, the partitions obtained according to the previous section, either in single or multiple folds, are processed to extract *collections* of examples. Then, each example is represented using one of the techniques introduced in section 3.2, that is, one among *short-time Fourier transform*, *mel frequency cepstral coefficients* and *wavelet scattering transform*. The extracted representations are optionally reduced and/or transformed to obtain a *solution* for the acoustic features required for the classification task. For instance, the *principal component analysis* or its kernel version (section 3.3) are employed for the purpose. The features obtained are then used to feed the classifier, either a *k-nearest neighbours* (section 3.4.1) or a *support vector machine* (section 3.4.2). Finally, the performance is assessed by calculating the error rate, the accuracy and the F-score (section 3.5). When relevant, a *confusion matrix* is also reported. All the results can be replicated with the associated software library developed.

It is remarked that the purpose of the analysis proposed here is to understand how to preserve the information of the audio in-pipe events when the signals are converted into acoustic features for classification purposes. Accounting for the issues discussed in section 3.2.1, we aim to compare a few possible solutions for feature extraction and also to provide a reference baseline for further developments on the same or other pipe-related datasets.

6.2.1 Data extraction and transformation chain

The analysis performed in this section aims to distinguish a class from the others in the same dataset. The input data are just short audio chunks extracted from the synthetic observations at points where the annotations indicate the presence of single classes. Disturbances are not included, and the background noise in the extracted chunks is not considered an additional class. In section 6.1, it has been shown how the synthesised dataset can be divided into balanced partitions where synthetic observations are separated guaranteeing non-overlapping source instances (*multi-folds*) or non-overlapping source observations (*single-fold*) between training and test set. The separated sets are further processed to obtain

the chunks required. Figure 6.8 illustrates the process implemented and the other data organisation steps described below.

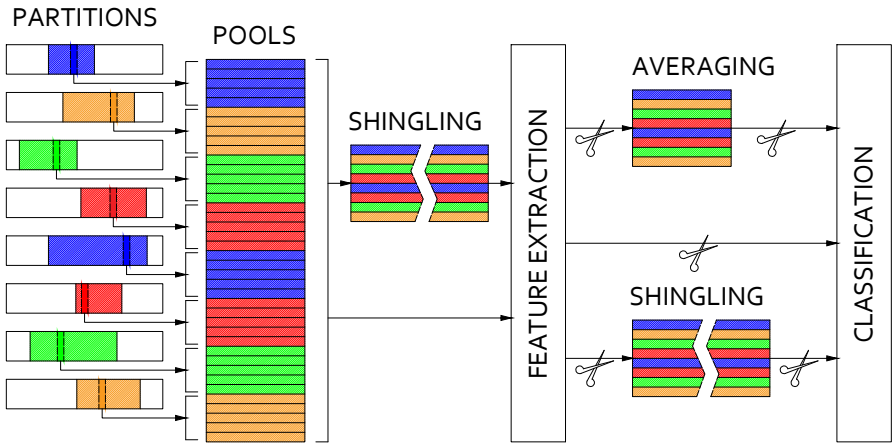


Figure 6.8: *Data extraction and transformation chain. The synthetic observations in the partitions are analysed to find the intervals where the classes are located. A number of pools of examples are then extracted from each interval and gathered in a collection of pools of examples. A collection includes a training set and a test set per each fold of the dataset. Examples in a pool can be used for feature extraction either directly or pre-shingled. The pools of features obtained can be used directly for classification or further operations, such as reduction, averaging and shingling, can be applied.*

The duration of the *atomic examples*, (e.g. $25ms$, $50ms$, $100ms$) is defined together with the number of examples per pool. A *pool of examples* is a set of N atomic examples of length L chosen to be either simply consecutive or overlapping for 50% of their length. A simple unitary step windowing function is applied at this stage. Each homogeneous interval found in a synthetic observation is processed to extract the desired number of non-overlapping pools or by tiling for the whole duration of the interval. If the partitions are balanced and with only one class per synthetic observation, a small number of short pools per observation can be extracted while leaving the resulting *collection* of examples still balanced. On the other hand, when the balance is lost in the extraction process, it can be reimposed with a random final selection that provides the same number of pools per class.

The examples in the extracted pools can be employed directly for future extraction (*unprocessed*) or further organised. In particular, a new single example can be

obtained by chaining together the examples in a pool (*pre-shingling*). Note that, when a pool is composed of an odd number of windows overlapping for 50% of their length, the exact original source signal of the pool can be rebuilt by skipping a window per each one chained starting from the first. Table 6.2 summarises the collections of examples used for classification purposes in the following sections.

Collection Id	Folds	Length (ms)	Pools per Interval	Examples per pool	Examples overlap (%)	Training pools per class	Test pools per class	Total training pools	Total test pools
C_{A0}	1	10	3	1	-	282	60	1692	360
C_{A1}	1	25	3	1	-	282	60	1692	360
C_{A2}	1	50	3	1	-	282	60	1692	360
C_{A3}	1	100	3	1	-	282	60	1692	360
C_{A4}	1	250	3	1	-	282	60	1692	360
C_{A5}	1	500	3	1	-	282	60	1692	360
C_{A6}	1	1000	3	1	-	282	60	1692	360
C_{B0}	1	50	3	19	50	282	60	1692	360
C_{B1}	1	50	3	19	50	282	60	1692	360
C_{B2}	1	50	3	19	50	282	60	1692	360
C_{B3}	1	50	3	19	50	282	60	1692	360
C_{C0}	4	500	3	1	-	390÷393	129÷132	2340÷2358	774÷792

Table 6.2: *Summary of the collections of examples used for classification. All the collections are extracted from the same dataset of 2000 synthetic observations reported in figure 6.2. Signal sample rate* $= 48kHz$.

From the pools of examples, unprocessed or shingled, the same number of *pools of features* are then extracted. In the unprocessed case, the pools of features obtained can be further transformed before classification. Pool-level operations can offer some benefits. For instance, pool-averaged features can reduce the effect of the noise, while shingling a pool of features in a single example potentially provides

a better description of longer sound structures. Further feature optimisations can be obtained in combination with reduction techniques described in section 3.3.

6.2.2 Performance of the STFT

For the first analysis performed, the features are obtained using the short-time Fourier transform. Although it is unfeasible to run full comprehensive multivariate optimisations for every parameter of the processing chain, meaningful tests can be performed to gain some understanding of convenient strategies for the extraction of the features. In particular, the STFT allows some useful insights into feature scaling and feature dimensionality.

Indicator	Classifier	Log	Pca	Collection/bins						
				C_{A0} 241	C_{A1} 601	C_{A2} 1201	C_{A3} 2401	C_{A4} 6001	C_{A5} 12001	C_{A6} 24001
F-score	KNN			0.353	0.397	0.472	0.503	0.592	0.564	0.553
		✓		0.542	0.494	0.517	0.581	0.569	0.564	0.603
			✓	0.372	0.389	0.453	0.489	0.55	0.564	0.564
		✓	✓	0.544	0.475	0.533	0.575	0.583	0.558	0.583
	SVM			0.458	0.458	0.497	0.517	0.539	0.536	0.528
		✓		0.572	0.575	0.672	0.722	0.711	0.736	0.767
			✓	0.414	0.372	0.436	0.503	0.522	0.536	0.544
		✓	✓	0.556	0.544	0.611	0.644	0.681	0.731	0.775

Table 6.3: *Average F-score performance of a 19-NN euclidean classifier and of a linear SVM classifier for windows of different lengths. The source collections (table 6.2) and the number of frequency bins of the STFT are reported on top. Total training pools = 1692, total test pools = 360. Performance is evaluated for simple STFT, for STFT log-scaled, STFT PCA-reduced and STFT log-scaled and PCA-reduced. Log-scaling is calculated as $log(1+|\hat{x}[i_f]|)$. The linear PCA is calculated by retaining 98% of the variance. When log-scaling is applied together with PCA, the former is calculated first.*

Table 6.3 reports the average STFT performance for the collections $C_{A0},...,C_{A6}$ of table 6.2. The F-score is calculated using a Euclidean 19-NN classifier and a linear SVM classifier.

The first line of each classifier reports the results obtained when the modulus of the STFT is used directly as feature set. Although a clear dependence on the length of the windows appears, the overall performance is quite poor for both KNN and SVM. Besides, given the large number of features used, this solution appears very inefficient. This poor performance certainly has more than one underlying reason. For instance, as mentioned in section 3.4.1, the curse of dimensionality generally entails worse performance for features carrying the same information in higher dimensions and, as pointed out by Vabalas [283], increasing the ratio between the number of features and the number of training examples generally makes the trained machine less accurate. Besides, as pointed out in section 3.2.1.3, the STFT does not provide very stable representations, especially for high-frequency components. One more issue (section 3.4.1) is related to the relative numeric magnitude of the features. Indeed, having subsets of feature values much bigger than others makes the latter irrelevant for classification, even when they carry important discriminative information.

6.2.2.1 Feature scaling

The last problem mentioned above can be tackled by non-linear feature scaling, that is, by compressing the largest values and magnifying the smallest. A simple and effective solution consists of using a logarithmic scale law, such as:

$$\hat{y}[i_f] = log(1 + |\hat{x}[i_f]|) \tag{6.1}$$

where $\hat{x}$ is the STFT. Adding 1 keeps the features always positive as for the modulus. The second line of both classifiers in table 6.3 shows the remarkable benefit achieved by applying this simple transformation. The benefit of using scaled features for audio processing seems to be recognised in the literature offering performance enhancement in applications such as speech [286] and music processing [287]. Log scaling is also an important component of bio-inspired features, and it is also adopted for MFCC and WST [79].

Note that a similar improvement can be obtained by applying alternative scaling

operations at the classifier level. For instance, if the Euclidean metric in the KNN classifier is replaced by the *cosine similarity*:

$$d(\boldsymbol{x},\boldsymbol{y}) = 1 - \frac{\boldsymbol{x}^T\boldsymbol{y}}{\|\boldsymbol{x}\|_2\|\boldsymbol{y}\|_2} = 1 - \frac{\sum_{i=1}^{N} x_i y_i}{\sqrt{\sum_{i=1}^{N} x_i^2}\sqrt{\sum_{i=1}^{N} y_i^2}}, \tag{6.2}$$

all the "distances" are normalised in the interval $[0,1]$, obtaining a scaling effect as for the log transformation.

Table 6.4 reports a comparison between the performances obtained by scaling at feature and classifier level for a KNN classifier. Note how applying a scaling operation at both the classifier and the feature level does not provide any further improvement and the performance remains similar to the one obtained by scaling only at the feature level.

Indicator	Classifier	Scale		Collection						
		F	C	C_{A0}	C_{A1}	C_{A2}	C_{A3}	C_{A4}	C_{A5}	C_{A6}
F-score	KNN			0.353	0.397	0.472	0.503	0.592	0.564	0.553
		✓		0.542	0.494	0.517	0.581	0.569	0.564	0.603
			✓	0.497	0.483	0.550	0.636	0.636	0.675	0.700
		✓	✓	0.575	0.531	0.525	0.575	0.536	0.561	0.569

Table 6.4: *Average F-score performance of a* 19*-NN classifier for different scaling solutions. Total training pools* $= 1692$*, total test pools* $= 360$*. The feature scaling (F) is obtained using equation* (6.1) *to scale the modulus, while classifier scaling (C) is obtained using equation* (6.2) *for the distance.*

6.2.2.2 Dimensionality reduction

The other main issue concerns dimensionality and efficiency of the representation. Table 6.3 shows that the performance increases with the length of the window, but, for instance, a set of features associated with any example of C_{A6} contains as many as 24001 variables.

In section 3.3.1, it has been shown how PCA can be used to reduce the dimen-

sionality by selecting only the principal components that carry most of the variance. Table 6.3 (third line KNN/SVM) reports the values of the performance for features reduced by retaining 98% of the variance. A comparison against the first line of the same table, however, reveals that no relevant benefit is gained in terms of performance. This is caused again by the unbalance in the magnitude of the STFT modulus. Indeed, when the STFT feeds the PCA directly, even removing only 2% of the variance yields a great reduction of the dimensionality of the features (greater than 99.5% for C_{A6}, table 6.5). This substantial reduction indicates that PCA discards most of the low-magnitude variables and the related information associated. Consequently, all the benefit gained in terms of dimensionality reduction is lost because of the loss of information. Similarly to the previous section, an improvement in the performance can be obtained by applying PCA after log-scaling the STFT. Table 6.3 (fourth line KNN/SVM) reports the performances for this case and shows how the dimensionality reduction can be achieved without losing the benefit associated with the non-linear scaling. For an optimal solution, however, it is important to determine the right number of retained components by assessing the trade-off between loss of information and reduced dimensionality.

	PCA	Log	Collection						
			C_{A0}	C_{A1}	C_{A2}	C_{A3}	C_{A4}	C_{A5}	C_{A6}
Features			241	601	1201	2401	6001	12001	24001
	✓		18	23	39	56	74	93	103
	✓	✓	72	175	314	526	776	945	1038

Table 6.5: *Number of features before and after PCA reduction retaining 98% of the variance. The log case is calculated by applying the scaling before calculating PCA.*

Figures 6.9 illustrate, for both classifiers, the variation of the F-score as a function of the number of principal components retained. In all the cases, the performance reaches its maximum value for no more than 100 principal components and either remains stable or decays afterwards. At their peak, the log-PCA features slightly outperform the full log representation showing that a compact set of features can greatly improve the efficiency without compromising the associated information.

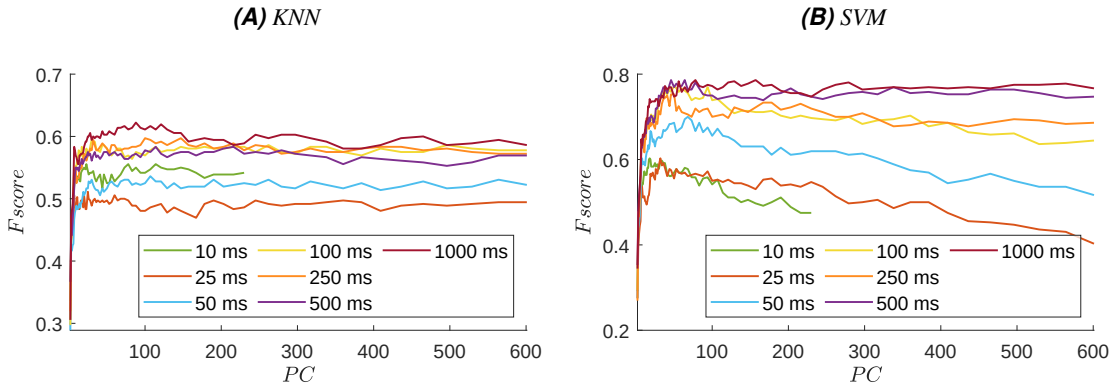


Figure 6.9: *Average F-score performance of the log-STFT as a function of the number of linear principal components retained. Collections $C_{A0},...,C_{A6}$. Total training pools = 1692, total test pools = 360. A) Euclidean 19-NN classifier. B) Linear SVM.*

Regarding the stability issue of the STFT mentioned above, it is important to remark that PCA does not offer any benefit. A simple alternative - to reduce the number of features and, at the same time, to attenuate potential instabilities - consists of grouping and averaging the log frequency components in frequency bins. Each bin, except the last one, is obtained with an equal number of consecutive frequency components. This approach, despite being very simple, has the advantage of being more robust to high-frequency deformations, as discussed in 3.2.1.3. Nevertheless, especially for a low number of bins, it penalises the information contained in the lower part of the spectrum.

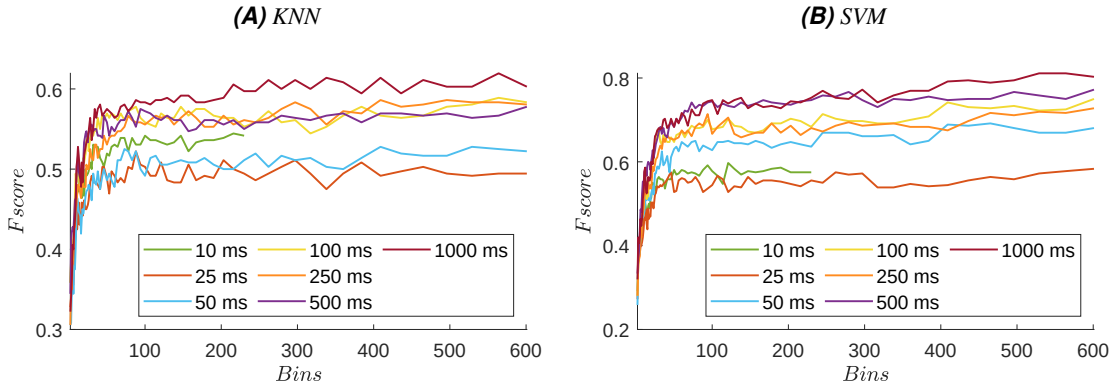


Figure 6.10: *Average F-score performance of the log-STFT with frequency components grouped and averaged in frequency bins of equal lengths. Performance values are reported as a function of the number of bins. Collections $C_{A0},...,C_{A6}$. Total training pools = 1692, total test pools = 360. A) Euclidean 19-NN classifier. B) Linear SVM.*

The resulting F-score as a function of the number of bins is reported in figure 6.10. Note that the curves saturate quickly and choosing no more than 100 bins yields just slightly worse results than more detailed solutions. The upper limit is given by the case reported in table 6.3 (second line KKN/SVM), which shows how the performances deteriorate when the number of bins grows too much. Note that the log-STFT grouping and log-STFT PCA return similar values, being the former slightly better for $250 \div 300$ bins and slightly worse for $100 \div 150$ bins. It is here remarked that, because of the equal size of the bins, the best grouping solution is determined as a trade-off between increased deformation stability and loss of low-frequency information.

6.2.2.3 Detailed performance of the STFT

This section briefly reports the performance per each class of the dataset when features are extracted using the STFT.

Collection	Classifier	Indicator	Average	Class					
				DW	SH	SI	TA	TO	WM
C_{B0}, C_{B1}, C_{B2}, C_{B3}	KNN	Err	0.369	0.567	0.433	0.192	0.383	0.125	0.513
		Acc	0.877	0.878	0.844	0.845	0.869	0.954	0.872
		Fsc	0.631	0.695	0.519	0.358	0.605	0.860	0.658
	SVM	Err	0.282	0.321	0.208	0.275	0.288	0.150	0.450
		Acc	0.906	0.924	0.907	0.865	0.901	0.950	0.889
		Fsc	0.718	0.793	0.698	0.534	0.699	0.850	0.702

Table 6.6: *Averaged performance indicators for the collections $C_{B0}, ..., C_{B3}$. Total training pools per collection = 1692, total test pools per collection = 360. Features are extracted by grouping and averaging the log-STFT components in 200 bins of equal length. Pools are pre-shingled in windows of* $500ms$.

Table 6.6 provides the indicator values averaged over the collections $C_{B0}, ..., C_{B3}$ of table 6.2. Pools are pre-shingled to return examples of $500ms$. This specific length is chosen since, from figures 6.10, SVM seems to provide better perfor-

mances than KNN and results for longer $1s$ windows are almost identical. The features are calculated by log-scaling the STFT and grouping the frequency components in 200 equal bins. The distance for KNN is calculated using the cosine similarity, since, despite the log-scaling applied to the STFT, it offers slightly better results than Euclidean metric. Figures 6.11 report the confusion matrices extracted from the classification of collection C_{B0}.

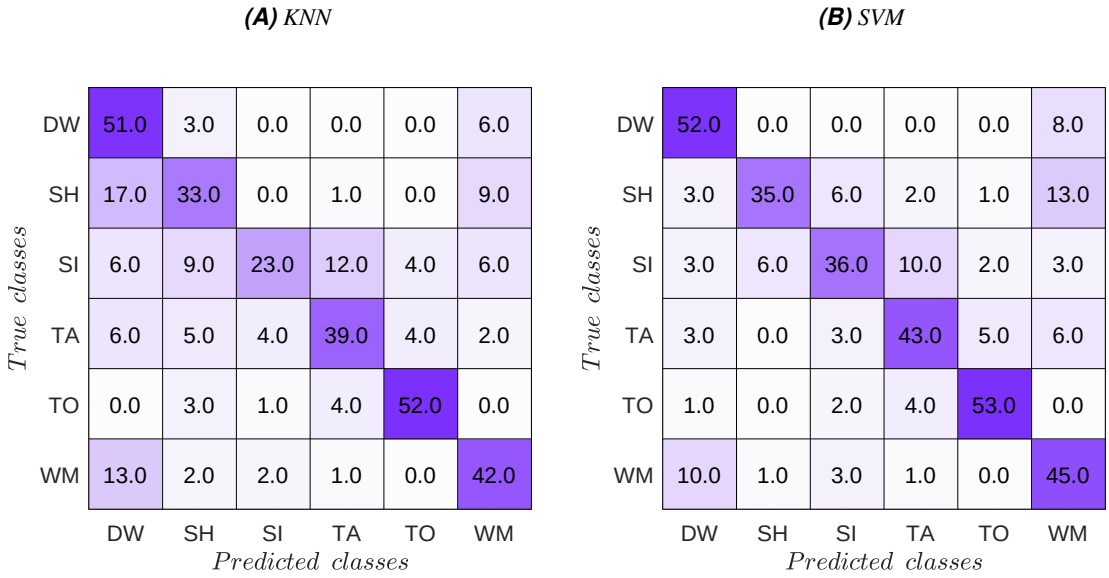


Figure 6.11: *Confusion matrix for the examples of collection C_{B0}. Total training pools = 1692, total test pools = 360. Features are extracted by grouping and averaging the log-STFT components in 200 bins of equal length. Pools are pre-shingled in windows of* 500*ms. A) Cosine 19-NN classifier. B) Linear SVM classifier.*

6.2.3 Performance of the MFCC

The second representation option analysed is based on the mel cepstral frequency coefficients described in section 3.2.2. For STFT, it has been shown how grouping frequency components in bins provides benefits to the performance. This transformation can be seen as a simple solution to obtain a stabler representation as discussed in section 3.2.1. Accounting for the differences, the filter bank used to calculate the MFCC delivers a similar operation. Because of the way the filter bank is built, however, lower frequency components are represented with higher resolution, while larger bins at higher frequencies make the representation more stable. Besides, the MFCC simulate the non-linearity of the auditory system by

taking the log of the averaged energies (section 3.2.2), thus implementing the feature scaling as seen for the STFT (section 6.2.2.1). In this section, we aim to optimise the MFCC representation to improve the performance of classification tasks on the proposed dataset.

6.2.3.1 Frequency range, number of filters, length of the window

Before attempting to find an optimal filter bank, a first test is performed to understand if the inspected spectra can be shrunk. Figures 6.12 report the performance of a linear SVM classifier for different values of the lowest and of the highest frequency boundaries of a 50-filters filter bank. The collections of examples used are $C_{A0}, ..., C_{A6}$ from table 6.2. The minimum and the maximum boundaries are chosen by dividing the whole frequency range $0 - 24kHz$ in 150 steps of equal mel length, according to equation (3.26) with $f_0 = 1000Hz$.

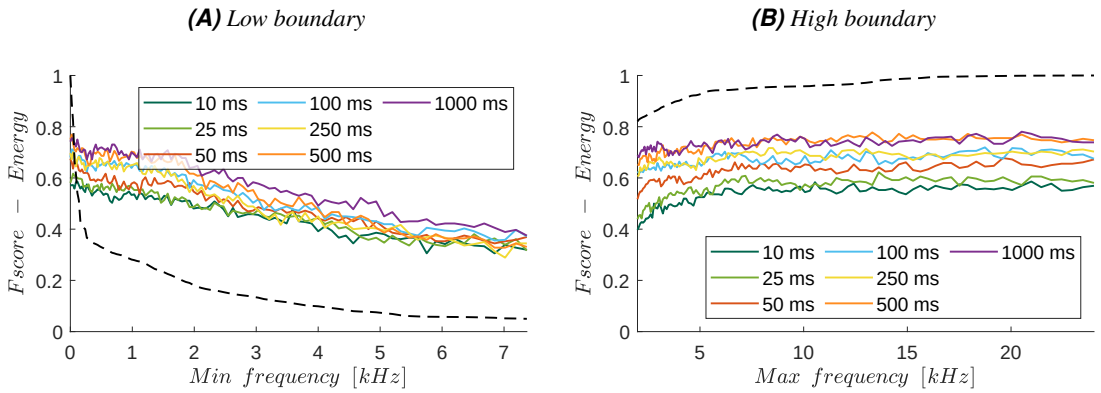


Figure 6.12: *Effects of the variation of the minimum and the maximum boundary frequencies for a 50-filters MFCC filter bank. Performances are obtained using a linear SVM classifier for different lengths of the windows. Boundary steps are obtained by dividing the mel scale into* 150 *equal intervals. Dashed lines report the normalised averaged energy captured over the whole training set. Examples collections:* $C_{A0}, ..., C_{A6}$ *from table 6.2. Total training pools = 1692, total test pools = 360. A) Performance for different values of the minimum boundary frequency. B) Performance for different values of the maximum boundary frequency.*

Figure 6.12A reports the average performance as a function of the lowest frequency boundary. The dashed line reports the normalised energy captured by the filter bank averaged over the whole training partition. Clearly, the lowest frequency components represent a relevant part of the signal energy, and the

performance diminishes roughly linearly when the lower boundary is moved toward higher frequencies. On the contrary, figure 6.12B shows how the highest part of the spectrum contains only a small portion of the energy (mainly associated with the background noise) and performance always saturates beyond $9 \div 10kHz$. Similar results can also be observed for KNN classifiers. Therefore, it is possible to cut the highest part of the spectrum out without significantly penalising the performance.

In figures 6.13, it is reported the performance as a function of the number of filters in the filter bank, with frequency range $0 - 12kHz$ and the number of MFCC coefficients equal to the number of filters. The collections used are the same as above. With only a few exceptions, regardless of the length of the window considered and for both classifiers, the performance increases with the number of filters. However, beyond a certain threshold, the benefit of a modest performance boost is hardly justified compared to the increased dimensionality of the features. Interestingly, as seen for the STFT, the actual value of the performance strongly depends on the length of the window used, which suggests that very low-frequency components play a key role in the classification. This is somewhat different from other audio classification tasks reported in literature where, as seen in section 3.2.2, no significant performance gain comes from windows longer than $25 \div 50ms$ [164],[288],[38, p. 27].

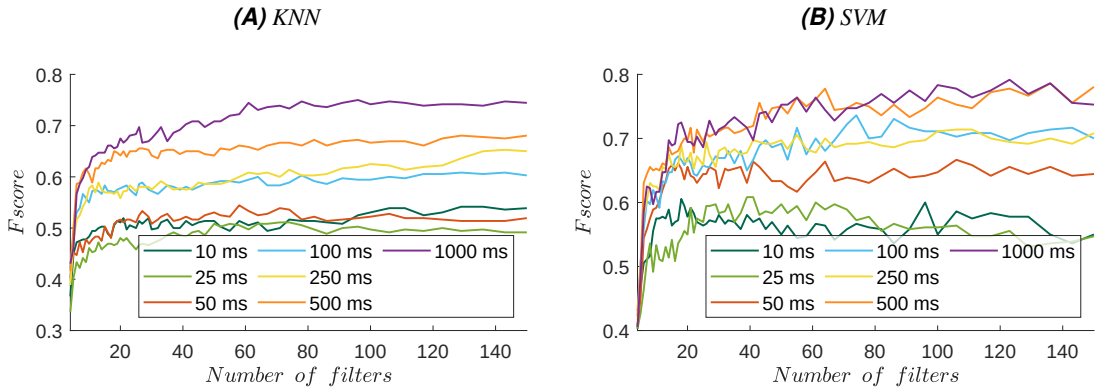


Figure 6.13: *Average performance as a function of the number of triangular filters and for different values of the window length. Windowing function: Chebyshev. The number of MFCC coefficients is equal to the number of filters. Collections of examples: $C_{A0},...,C_{A6}$ from table 6.2. Total training pools = 1692, total test pools = 360. A) Cosine 19-NN classifier. B) Linear SVM classifier.*

6.2.3.2 Low-pass filter selection

Recalling section 3.2.1, the feature extraction process can be tailored either to increase invariance or to preserve the information contained in the signal. The trade-off is mainly determined by the length of the averaging filter, which, for MFCC, is given by the windowing function ϕ (section 3.2.2).

From the results reported in figures 6.13, SVM seems to offer slightly better performance than KNN, and we now refer mostly to the former. Besides, windows longer than $500ms$ provide no additional benefit and going beyond $1s$ might be undesired in the case of real-time machines.

Assuming $500ms$ to be the length of the audio chunk to be converted into features, two different extreme-case options appear relevant for the determination of the above-mentioned trade-off. The first option, the one used for figures 6.13, aims to achieve feature invariance by applying a single window over the entire $500ms$ interval. The second one aims to preserve information by processing consecutive shorter windows and merging together the resulting features.

For a reliable comparison, both options need to be analysed by referring to the same collection of examples, which are chosen to be $C_{B0}, ..., C_{B3}$ from table 6.2. By extracting atomic examples of $50ms$ organised in pools of 19 with 50% overlap, as described in section 6.2.1, it is possible to reuse the same set of examples for the two different cases. The continuous $500ms$ signal is obtained by alternate pre-shingling, with a windowing function $500ms$ long applied before feature extraction. Conversely, for the second solution, a windowing function of $50ms$ is applied to each of the 19 examples of the pools and the shingling operation is performed after feature extraction[1]. Note that, in terms of information contained, the two source signals are equivalent, in the sense that one can be obtained from the other and vice-versa.

In section 3.2.1, it has been mentioned that the shape of the windowing function can affect the performance since windowing inevitably introduces frequency arte-

[1] With reference to the processing chain of figure 6.8, shingling or averaging are both possible strategies to merge the features of the 19 examples of the pool. Although both solutions yield similar results, some interesting remarks about the differences are reported at the end of the section.

facts mainly associated with the lateral lobes of the spectrum. Table 6.7 shows the F-score performance of a linear SVM classifier for the two window lengths defined above, for three different windowing functions, and for the four collections of examples mentioned above. Both the number of filters and the number of coefficients of the MFCC representation are set to 60.

Collection	Window / Length					
	Kaiser 50ms	Gauss 50ms	Chebyshev 50ms	Kaiser 500ms	Gauss 500ms	Chebyshev 500ms
C_{B0}	0.742	0.778	0.758	0.789	0.781	0.769
C_{B1}	0.714	0.764	0.767	0.756	0.739	0.722
C_{B2}	0.739	0.792	0.808	0.792	0.794	0.772
C_{B3}	0.703	0.753	0.750	0.742	0.739	0.722

Table 6.7: *Average F-score performance of the MFCC as a function of the low pass filter length and of the windowing function. Results are obtained for collections $C_{B0}, ..., C_{B3}$ of table 6.2. Total training pools per collection = 1692, total test pools per collection = 360. Number of MFCC filters: 60. Number of MFCC coefficients: 60. Classifier: linear SVM.*

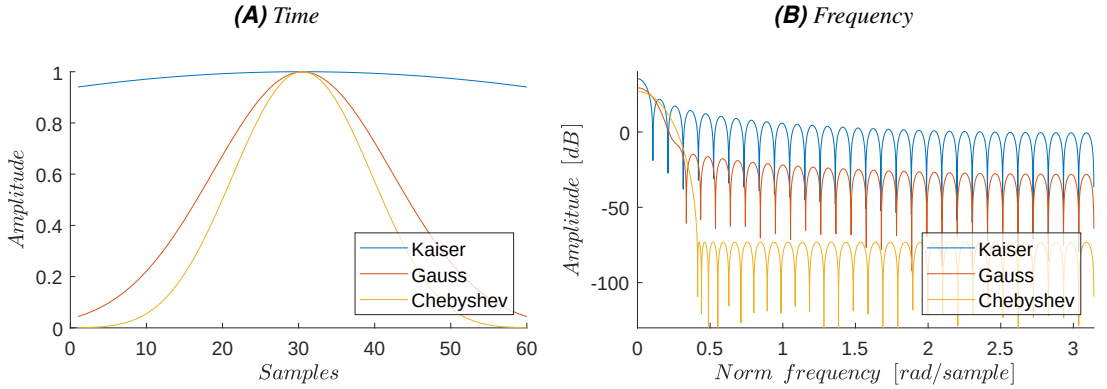


Figure 6.14: *Representation in time and frequency of the Kaiser, the Gauss and the Chebyshev windowing functions.*

A quick analysis of table 6.7 and figure 6.14 [163] shows that, when the artefacts introduced by the lateral lobes are low, the performance is higher for shorter windows (Chebyshev: $0.771@50ms \rightarrow 0.746@500ms$ on average). On the contrary, the spectrum of the Kaiser window shows important lateral lobes and the

performance improves with the length (Kaiser: $0.724@50ms \rightarrow 0.769@500ms$ on average). The Gauss window, whose lateral lobes are less pronounced than the Kaiser but more than the Chebyshev, returns a similar performance for both the window lengths (Gauss: $0.771@50ms \rightarrow 0.763@500ms$ on average). Hence, when choosing the windowing function, avoiding high lateral lobes is important only when windows are short. Overall, the average values of the performances are comparable for the $50ms$ and the $500ms$ solutions, being the latter just 0.5% higher than the former over the full unitary scale. This suggests that the increased representation invariance compensates for the loss of information.

It is important to remark, however, that the performances reported in table 6.7 are obtained with two very different numbers of coefficients, namely 60 for the $500ms$ window and 19×60 for the $50ms$ one. Hence, a better comparison can be obtained by repeating the same test but reducing the features to the same number using linear PCA. Table 6.8 reports the results obtained retaining only the first 50 principal components for both cases.

Collection	Window / Length					
	Kaiser 50ms	Gauss 50ms	Chebyshev 50ms	Kaiser 500ms	Gauss 500ms	Chebyshev 500ms
C_{B0}	0.756	0.775	0.769	0.822	0.828	0.797
C_{B1}	0.714	0.733	0.772	0.781	0.792	0.781
C_{B2}	0.694	0.758	0.744	0.786	0.808	0.794
C_{B3}	0.700	0.739	0.725	0.747	0.761	0.736

Table 6.8: *Performance of the solutions reported in table 6.7 obtained by limiting the number of features to* 50 *using linear PCA reduction.*

When the dimensionality of the features is the same, filtering by using longer windows provides better performance, and table 6.8 shows an average difference slightly below 5% over the full unitary scale. Note that the long Gauss window returns an average performance that is 1% better than the Kaiser and 2% better than the Chebyshev.

For completeness, it is here remarked that an alternative solution for the short windows consists of post-averaging the pools instead of post-shingling. This solution directly returns 60 features and results similar to those reported in table 6.7. This circumstance is mainly related to the noise cancellation effect provided by the averaging operation on the features. Nevertheless, since features are cleaner, when PCA is applied, dropping a few principal components means also dropping a greater portion of relevant information. Indeed, reducing the principal components from 60 to 50 yields similar results to those reported in table 6.8.

The results above, suggest that working with longer windows and selecting appropriate windowing functions can improve the overall performance, meaning that preserving feature invariance is worth sacrificing part of the signal information. Besides, as shown in section 3.2.3.4, featurization based on wavelets scattering transform allows the recovery of the discarded information, and the adoption of this solution further limits the benefit of working with shorter windows.

6.2.3.3 Filter bank optimisation

The definition of the MFCC features can be further improved by determining an optimal solution for the filters of the filter bank. If the maximum and the minimum frequencies are chosen as explained above, the optimisation concerns the number F of triangular filters and the related frequency bands. It is firstly remarked that attempting to find the filter edges by optimising them individually is unlikely to provide a valuable solution since it translates into the maximisation of a cost function of F variables with a large number of local maxima. Applying techniques such as gradient descent scarcely adapts to the discrete nature of the problem and generally leads to local maxima where the performance is lower than what is achievable with standard MFCC. Besides, as seen in section 3.2.1, the distribution of filters with larger grouping toward higher frequencies is likely to be more robust by introducing stability against issues such as time-warping deformations. Since stability also means better generalisation, it is important to preserve this property in the optimisation process.

A viable solution consists of keeping the filter edges equally spaced in the mel

scale while trying to optimise the number of filters F and the conversion law between linear frequency f and mel scale $\bar{f}$. Therefore, we seek to improve the performance by searching sub-optimal values for F and for the constants C and f_0 in:

$$\bar{f} = \frac{f_0}{log(C)} \times log\left(1 + \frac{f}{f_0}\right), \tag{6.3}$$

where standard MFCC coefficients are generally calculated assuming $C = 2$ and $f_0 = 1000Hz$ [176]. Besides, if $\bar{f}_{C_1} \to f$ and $\bar{f}_{C_2} \to f$ are two different values of the mel scale calculated for two different C (with the same f_0) but corresponding to the same linear frequency f, then $log(C_1)\bar{f}_{C_1} = log(C_2)\bar{f}_{C_2}$ and $log(C_1)\bar{f}_{C_1}/k = log(C_2)\bar{f}_{C_2}/k$, which means that the edges of the filters in the linear frequency do not depend on the value of C. Therefore, the optimisation concerns only F and f_0, while C can be assumed equal to 2. Note that it is not convenient trying to optimise the Q factor of the filter bank. Indeed, as shown in section 3.2.2, mel filters are only approximately Q constant (the approximation is invalid at low frequencies) and trying to impose the same Q for all the filters would mean a ratio of the last to the first frequency edge roughly proportional to Q^{F-1}.

Implementing a supervised algorithm requires the definition of a *cost function* and, for this task, we refer to SVM since it seems to offer slightly better performance than KNN. In section 3.4.2.3, it has been shown that multi-class classification can be performed by introducing a *binary loss*:

$$l_b(\boldsymbol{x}, j, k) = -g(\bar{y}_{jk}, s_j(\boldsymbol{x})), \tag{6.4}$$

where j identifies the binary classifier, k the class, $\boldsymbol{x}$ the example to classify, and $s_j(\boldsymbol{x}) = \boldsymbol{w}_j^T\boldsymbol{x} + b_j = \sum_{i=1}^{N} \alpha_{ji} y_{ji} \boldsymbol{x}_{ji}^T \boldsymbol{x} + b_j$ the related score. Assuming $g(y,s)$ as the *hinge function* (equation (3.101)) and defining $\bar{y}_{jk} = \{-1, 0, 1\}$ as from section 3.4.2.3, the binary loss l_b is always ≤ 0 and the magnitude of the negative values give a measure of the classification uncertainty for the binary classifiers. Averaging the binary loss over all the binary classifiers as from (3.100), it is possible to obtain the *negative loss* $l(\boldsymbol{x}, k)$, that is a measure of the classification uncertainty for a

specific class [226]. Using the definitions above, the *classification margin* for the example $\boldsymbol{x}$ can be defined as [87]:

$$m(\boldsymbol{x}) = l(\boldsymbol{x}, k_T) - max\big(l(\boldsymbol{x}, k_F)\big), \tag{6.5}$$

that is the difference between the negative loss of the true class and the maximum negative loss among the false classes. The classification margin provides a measure of the robustness of the classification, being a larger margin generally associated with better performances. Finally, averaging the margin over the whole training set[2] provides a scalar function usually referred as *classification edge* [87]:

$$E(F, f_0) = \frac{\sum_{n=1}^{N} m(\boldsymbol{x_n})}{N}. \tag{6.6}$$

The classification edge can be seen as a function of the set of features chosen, and consequently, as a function of F and f_0. Nevertheless, referring to figure 6.13, it is reasonable to assume that using the definition (6.6) as a cost function would generally return the maximum number of filters even when the benefit of higher feature dimensionality is marginal or negligible. Therefore it is necessary to introduce a *regularisation term* that penalises the cost when the number of filters F increases without any relevant benefit, that is:

$$\Lambda(F, f_0) = E(F, f_0) - w_r F. \tag{6.7}$$

A reasonable value for the regularisation multiplier w_r can be assigned by referring again to figure 6.13 and considering roughly negligible the increase in the performance beyond the corner points of the plots. With this assumption, the value of the classification edge is sampled for $f_0 = 1000Hz$ and for a few values of F between $F_{max}/2$ and F_{max}. The sampled values $E(F_1), E(F_2), ...E(F_H)$, are then linearly interpolated as the least-squares solution of the problem [289]:

[2] Since the dataset is balanced, no prior weighs are necessary when the (6.6) is calculated.

$$\begin{bmatrix} F_1 & 1 \\ F_2 & 1 \\ \vdots & \\ F_H & 1 \end{bmatrix} \begin{bmatrix} b \\ a \end{bmatrix} \approx \begin{bmatrix} E(F_1) \\ E(F_2) \\ \vdots \\ E(F_H) \end{bmatrix}. \tag{6.8}$$

Assuming $w_r = b$, the negligible increment of performance shown in figure 6.13 does not determine any increment of the cost function $\Lambda(F, f_0)$.

Algorithm 5 Mel scale and number of filters optimisation

Input: $H, M, [F_{min}, F_{max}], [f_{0_{min}}, f_{0_{max}}], \{\boldsymbol{r}_l\}$

Output: F, f_0

$w_r \leftarrow$ from $E(F_1, f), ..., E(F_H, f) \;\big|\; F_h \in [F_{max}/2, F_{max}], f = 1000$

$\{l_m\} = 1$ ▷ Init radii to the first $\forall\, m \in [1 : M]$

$\{\boldsymbol{p}_m\} \leftarrow$ Random $\boldsymbol{p}_m \in ([F_{min}, F_{max}], [f_{0_{min}}, f_{0_{max}}])$

$\{\Lambda(\boldsymbol{p}_m)\}$ ▷ M initial cost values @ $\{\boldsymbol{p}_m\}$

while $length(\{\boldsymbol{p}_m\}) > 1 \;\;\&\;\; min\{l_m\} < L$ **do**

  for each $\boldsymbol{p}_m \;|\; l_m < L$ **do**

    $\{\boldsymbol{p}_{m\Delta}\} = (F_m \pm r_{F_{lm}}, f_m), (F_m, f_m \pm r_{f_{lm}})$ ▷ 4 neighbouring points of $\boldsymbol{p}_m$

    $\{\Lambda(\boldsymbol{p}_{m\Delta})\}$ ▷ Cost values of the 4 neighbours

    if $\Lambda(\boldsymbol{p}_{m\Delta}) > \Lambda(\boldsymbol{p}_m)$ **then**

      $\boldsymbol{p}_m \leftarrow \boldsymbol{p}_{m\Delta}$ ▷ Update the center point

      $\Lambda(\boldsymbol{p}_m) \leftarrow \Lambda(\boldsymbol{p}_{m\Delta})$ ▷ Update the center reference cost

      $l_m \leftarrow l_m - 1$ ▷ Increase radius if $l_m > 1$

    else

      $l_m \leftarrow l_m + 1$ ▷ Reduce radius

    end if

  end for

  $\{\boldsymbol{p}_m\} \leftarrow \{\boldsymbol{p}_m\} - \boldsymbol{p}_{m^*} \;\big|\; \Lambda(\boldsymbol{p}_{m^*}) = min\{\Lambda(\boldsymbol{p}_m)\},\; length(\{\boldsymbol{p}_m\}) > 1$

end while

$(F, f_0) \leftarrow \boldsymbol{p}_m$

return (F, f_0)

Although the application of gradient ascent strategies remains unfeasible, the cost function defined above depends only on two variables and its optimisation is certainly easier than dealing with each single filter edge in the filter bank. The solution proposed to maximise the cost function is inspired to *pattern search* [84] and *simulated annealing* [85], and it is summarised by algorithm 5.

Once the cost function $\Lambda(F, f_0)$ has been defined, the algorithm proposed starts with a set of M random points $\{\boldsymbol{p}_m\} \equiv [(F_1, f_{01}), ..., (F_M, f_{0M})]$ determined in the range $F_m \in [F_{min} : F_{max}] = [10 : 150]$ and $f_{0m} \in [f_{0_{min}}, f_{0_{max}}] = [1, 10000]$. Further expansion of these domains does not seem to provide any benefit. A set of L discrete and descending search pattern radii $\{\boldsymbol{r}_l\} \equiv [(r_{F_1}, r_{f_{01}}), ..., (r_{F_L}, r_{f_{0L}})]$, with $r_{F_L} = r_{f_{0L}} = 0$, is also defined beforehand. The initialisation is completed by calculating the cost for each random point $\boldsymbol{p}_m$, hence providing the values $\{\Lambda(\boldsymbol{p}_m)\}$ for the first step of the iterative search. The iterations are aimed to move the random points $\{\boldsymbol{p}_m\}$ towards better performances and gradually discard those points that yield lower cost.

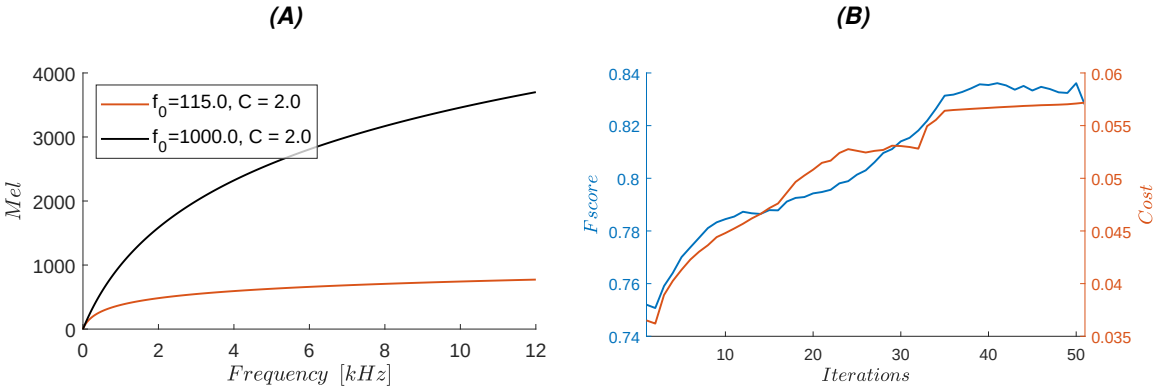


Figure 6.15: *Mel scale and number of filters optimisation. Length of the training examples: 500ms. Low-pass filter window: Gaussian. Collection used: C_{B0} with alternate pre-shingling (table 6.2). A) Comparison between standard and optimised Hz-mel conversion law. B) Results of the optimisation algorithm obtained using 50 initial random points. Average F-score obtained using 360 test pools from C_{B0} .*

At each iteration step, a cross pattern of four neighbouring points $\{\boldsymbol{p}_{m\Delta}\}$ around each $\boldsymbol{p}_m$, $(F_m \pm r_{F_{lm}}, f_{0m})$ and $(F_m, f_{0m} \pm r_{f_{0lm}})$, is selected to assess the direction toward which the cost increases. If a neighbour $\boldsymbol{p}_{m\Delta}$ offers better performance, it substitutes the centre point $\boldsymbol{p}_m$ for the next iteration. Otherwise, the current $\boldsymbol{p}_m$ is

kept, but the neighbouring radius is reduced ($l_m \leftarrow l_m + 1$) to investigate further closer points. If the point $\boldsymbol{p}_m$ changes, whether possible, the selected radius is increased by one step ($l_m \leftarrow l_m - 1$). When the neighbouring radius reaches the last step $(r_{F_L}, r_{f_L}) = (0,0)$, no further evaluation on the neighbours of $\boldsymbol{p}_m$ is performed since it is assumed that a local maximum has been found. At each iteration, the worst performing point is removed from $\{\boldsymbol{p}_m\}$ and the list of points to check is shrunk by one. The algorithm stops when a single point remains on the list, and the related radius is null $(0,0)$.

Figure 6.15B illustrates the improvement of the average performance at $\{\boldsymbol{p_m}\}$ obtained with the algorithm described above for 50 initial random points. Examples of $500ms$ with a Gaussian low pass filter have been considered. Clearly, the performance saturates beyond 35 iterations, meaning that all the remaining $\boldsymbol{p_m}$ have reached their local maximum. To check against overfitting, the average F-score is evaluated on the correspondent test set. The F-score performance improves with the cost and goes from roughly 0.75 to almost 0.84 obtained for $F = 60$ $f_0 = 115Hz$. Over the full unitary scale, this value is roughly 6% higher than the correspondent non-optimised value reported in table 6.7 (C_{B0}, 500ms, Gauss, no PCA). Note that each iteration step is supervised only with data belonging to the train set while the test set classification performance (average F-score) is calculated only for reference.

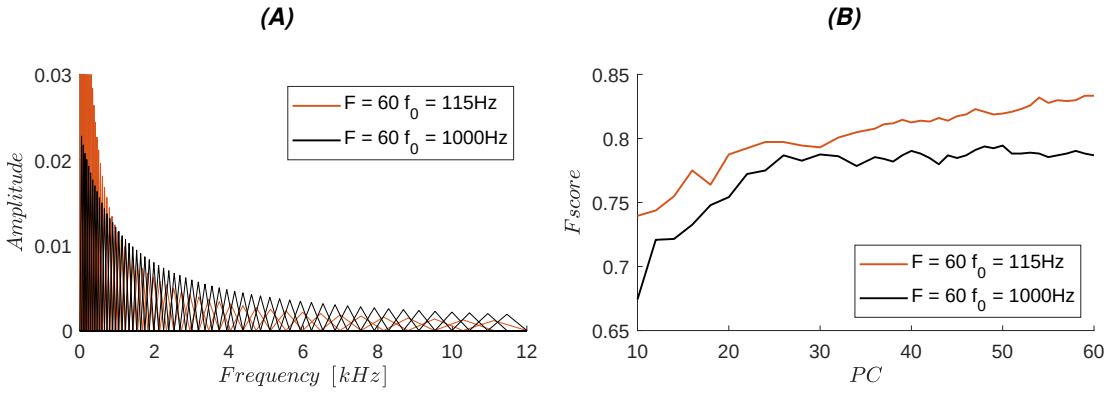


Figure 6.16: *A) Standard (black) and optimised (red) mel filter bank. Number of filters = 60. The optimised filters provide higher resolution at lower frequencies and lower resolution at higher frequencies. B) Average F-score of the standard (black) and of the optimised (red) MFCC as a function of the number of retained linear principal components. Standard representation saturates beyond 30 PC while optimised representation increases steadily. Results averaged over $C_{B0}, ..., C_{B3}$ (table 6.2, 360 test pools per collection).*

Figure 6.15A compares the Hz-mel conversion law before and after the optimisation, showing that the optimised filter bank exhibits much higher resolution at low frequencies and a smaller number of filters at high frequencies. Again, this confirms the importance of low-frequency components, their role in the discrimination of different classes, and the need to use windows much longer than $20 \div 50ms$. The resulting filter bank is illustrated in figure 6.16A. Note that a much lower corner frequency f_0 means that the Q-constant approximation is valid over a larger range of frequencies. Finally, a lower f_0 also means that relevant low-frequency information is spread on more features while high-frequency noise tends to be collected by a minor number of larger filters. As a consequence, using PCA, the performance of the optimised solution steadily increases with the number of retained principal components while the standard MFCC roughly saturates beyond 30 (figure 6.16B).

Collection	Std. $F = 60$, $f_0 = 1000Hz$, $PC = 50$			Opt. $F = 60$, $f_0 = 115Hz$, $PC = 55$		
	Kaiser 50ms	Gauss 50ms	Chebyshev 50ms	Kaiser 500ms	Gauss 500ms	Chebyshev 500ms
C_{B0}	0.822	0.828	0.797	0.858	0.861	0.817
C_{B1}	0.781	0.792	0.781	0.842	0.814	0.806
C_{B2}	0.786	0.808	0.794	0.842	0.831	0.825
C_{B3}	0.747	0.761	0.736	0.792	0.808	0.778

Table 6.9: *Average F-score comparison between standard and optimised MFCC. 50 PC are retained for the former while 55 for the latter. The different number of PC is dictated by the different saturation behaviour illustrated in picture 6.16B. Number of filters $F = 60$. Total training pools per collection = 1692, total test pools per collection = 360. Average standard performance $= 0.786$. Average optimised performance $= 0.823$.*

Table 6.9 reports the comparison between the standard MFCC reduced to the first 50 PC (see also table 6.8) and optimised MFCC reduced retaining the first 55 PC. The last 5 components are discarded for noise reduction. In both cases, the number of filters is set to 60. Over the full unitary scale, the average F-score of the optimised solution (0.823) is almost 4% higher than the other (0.786) and more than 6% higher if compared to the one not reduced of table 6.7 (0.7597).

6.2.3.4 MFCC detailed performance and class clusters

This section provides a detailed overview of the classification when the optimisations reported above are applied together. Therefore, features are extracted by assigning $f_0 = 115Hz$ and $C = 2$ in the conversion law (6.3), with 60 filters between 0 and $12kHz$, and using linear PCA to remove the last 5 principal components. The Gauss window is chosen for the low pass filter.

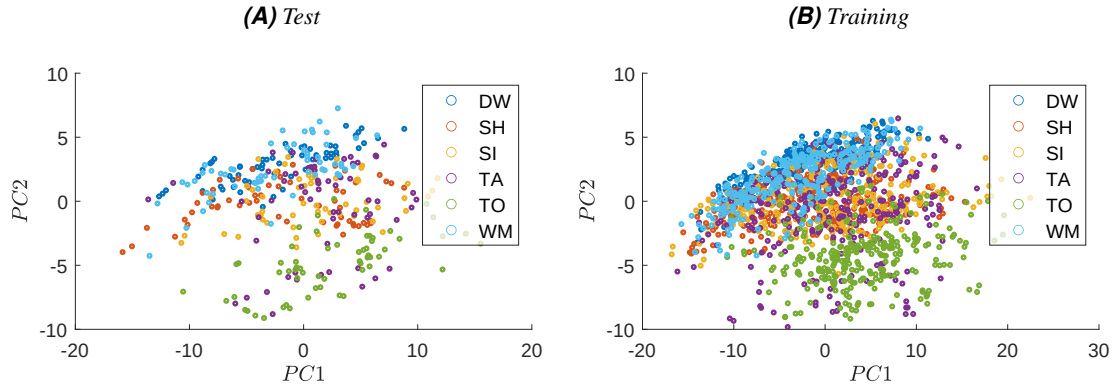


Figure 6.17: *Linear PCA scatter plot. Representation of the first two principal components for the collection C_{B0} of table 6.2. Window: Gauss, $f_0 = 115Hz$, $filters = 60$, $f_{max} = 12kHz$. A) Test set. B) Training set.*

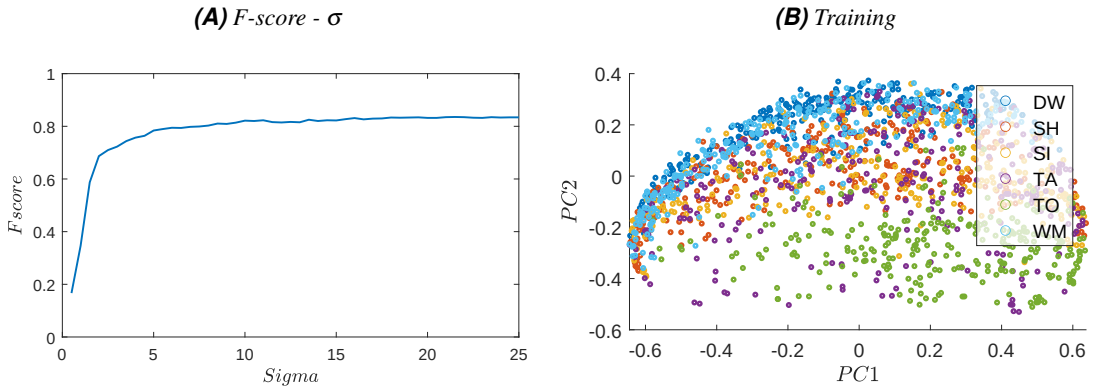


Figure 6.18: *Performance of the Gaussian-kernel PCA. Window: Gauss, $f_0 = 115Hz$, $f_{max} = 12kHz$, $filters = 60$, collection: C_{B0} (table 6.2, training pools = 1692, test pools = 360). A) Average F-score as a function of the parameter σ of the Gaussian kernel. B) Scatter plot of the first two principal components for $\sigma = 13$.*

Figures 6.17 provide an overview of the clusters given by the two first principal components for the training and the test set of C_{B0}. Although only in 2 dimensions,

a certain degree of clustering appears. Classes, however, are certainly divided into sub-clusters, meaning that a large number of different class instances should be included in the dataset to increase the robustness of the classification against unknown sound sources. In section 3.3.1.2, it has been mentioned that non-linear kernels applied to PCA can provide further improvement. This fact, however, is strongly dependent on the kernel used. Figures 6.18, for instance, show that the application of a Gaussian kernel with a similar number of retained components does not provide any additional benefit, being the maximum value of the performance for different values of σ very similar to the best performance achievable with linear PCA.

Collection	Classifier	Indicator	Average	Class					
				DW	SH	SI	TA	TO	WM
C_{B0}, C_{B1}, C_{B2}, C_{B3}	KNN	Err	0.303	0.300	0.333	0.175	0.329	0.300	0.383
		Acc	0.899	0.942	0.897	0.865	0.869	0.932	0.888
		Fsc	0.697	0.847	0.700	0.466	0.582	0.814	0.676
	SVM	Err	0.170	0.050	0.275	0.242	0.258	0.088	0.108
		Acc	0.943	0.988	0.922	0.910	0.923	0.962	0.956
		Fsc	0.830	0.963	0.774	0.722	0.775	0.882	0.865

Table 6.10: *Performances of the optimised MFCC representation averaged over collections $C_{B0},...,C_{B3}$ of table 6.2. Window: Gauss, $f_0 = 115Hz$, $filters = 60$, $f_{max} = 12kHz$. Total training pools per collection = 1692, total test pools per collection = 360.*

Table 6.10 provides the optimised averaged performance indicators over collections $C_{B0},...,C_{B3}$ of table 6.2. A comparison against the equivalent table for STFT (table 6.6) reveals that SVM returns a F-score improvement of 11.2% over the full unitary scale. Error rate and accuracy also improve accordingly going from 0.282 to 0.170 for the former, and from 0.906 to 0.943 for the latter. Improvements for KNN are more contained, with the F-score 6.6% higher over the full unitary scale.

Finally, as for figure 6.11, figure 6.19 illustrates the confusion matrix of the optimised MFCC representation for the collection C_{B0} and for both KNN and

SVM. Note that all the classes indicate better classification, with the most evident improvements related to the worst performing classes of the SFTF.

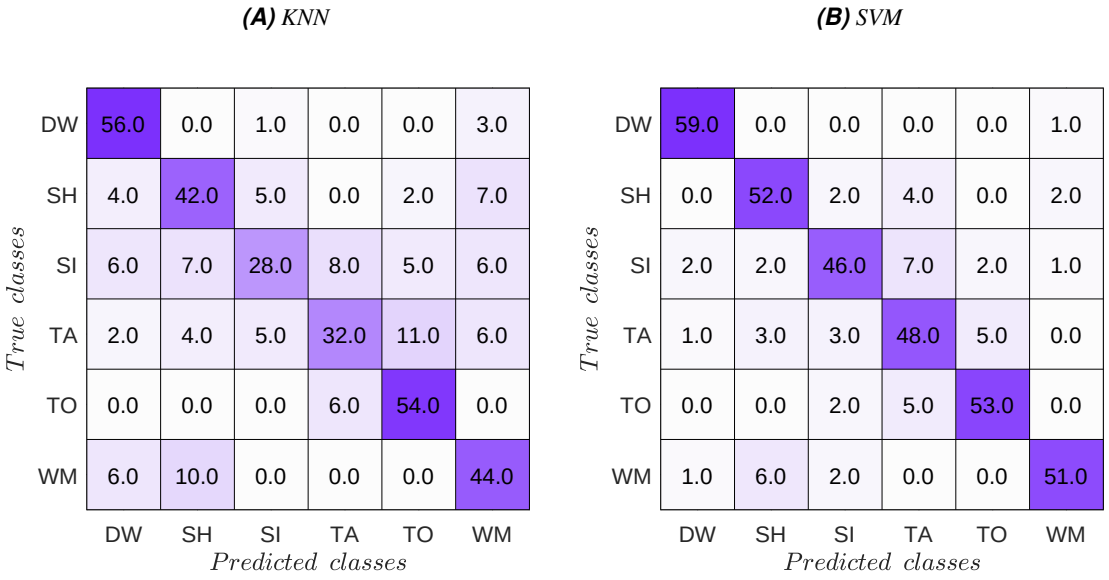


Figure 6.19: *Performance of the optimised MFCC for the collection C_{B0} of table 6.2. Window: Gauss, $f_0 = 115Hz$, $filters = 60$, $f_{max} = 12kHz$, Principal components $= 55$. A) Cosine 19-NN classifier. B) Linear SVM classifier.*

6.2.4 Performance of the wavelet scattering transform

In the previous sections, it has been shown that working with low-pass filters implemented with longer windows provides better performance by increasing the representation invariance and by capturing low-frequency components. The drawback of this approach is the loss of information that inevitably arise when some form of averaging operation is performed. As discussed in section 3.2.3.4, one of the most interesting properties of the wavelet scattering transform (WST) is the capability to recover the discarded information by stacking additional layers of wavelet processing. Hence, in this section, this third representation and some of the optimisations found for the STFT and the MFCC are applied together to further improve the performance of the extracted features. In the following, we mainly refer to the software implementation provided by *Scatnet* [290].

6.2.4.1 Optimisation of the filter banks

As for STFT and MFCC, the inspected spectrum can be reduced to $0-12kHz$ with negligible repercussions on the performance. Moreover, as shown below, this restriction of the inspected spectrum can even be beneficial, and the considerations reported in section 6.2.2.1 about feature scaling remain valid. Indeed, similarly to the other cases, log-scaling the WST features boosts the performance without any drawback. Normalisation of the coefficients as described in section 3.2.3.4 is also empirically proven to be beneficial.

When features are extracted using the WST, the amount of signal energy - and the associated information - diminishes moving from one layer to the next [79]. Hence, apart from layer zero, which merely returns the value of the input signal $x[n]$ time-averaged by the low-pass filter $\phi_0[n]$ ($x[n] \star \phi_0[n]$), a proper definition of the first layer is crucial. Although it is certainly necessary to perform a direct optimisation, as shown in section 3.2.3.3, the mathematical operation implemented by the first layer of the WST is very similar to the one associated with MFCC. Therefore, the results obtained for the MFCC can be used as a starting point for the optimisation of both the averaging filter $\phi_1[n]$ and the filter bank $\{\psi_{1,:}[n]\}$.

As for the MFCC, the low pass filter $\phi_1[n]$ is implemented using a Gaussian window covering the whole $500ms$ length. In Scatnet, however, the actual low-pass filter is not determined directly by choosing the length of a given window. Instead, it is calculated from the properties of the mother wavelet $\psi[n]$, namely the number of filters (or wavelets) per octave Q and the reciprocal of the octave bandwidth B. Named L the number of samples of the windows, h the index of the layer ($h \in [1:H]$), J_h the maximum wavelet scale, and being the scaling index $j \in [0:J_h-1]$, the largest wavelet in the time domain is:

$$\psi_{J_h-1}[n] = 2^{-\frac{J_h-1}{Q_h}} \psi_h(2^{-\frac{J_h-1}{Q_h}}[n]), \tag{6.9}$$

while the related normalised frequency bandwidth is:

$$BW_{J_h-1} = 2^{-\frac{J_h-1}{Q_h}}(1-2^{-\frac{1}{B_h}})\pi. \tag{6.10}$$

The correspondent time duration is calculated as [291]:

$$T_h = 2^{\frac{J_h-1}{Q_h}} 4B_h. \tag{6.11}$$

From (6.11), J_h is found as:

$$J_h = 1 + round\left(Q_h log_2 \frac{L}{4B_h}\right), \tag{6.12}$$

that is, imposing the duration of the largest wavelet to be as close as possible to the duration of the desired window L. The low-pass filter is then chosen to have the same duration and bandwidth as for $\psi_{J_h-1}[n]$ but centred at null frequency [3].

The mother wavelet here adopted is a *Morlet* wavelet (section 3.2.3.1), that is an analytic wavelet based on a Gaussian window (equation (3.39))[4]. A Gaussian window is also used for the low-pass filter. With reference to equations 3.40, the low-pass Gaussian filter remains defined by retrieving the value of $\sigma_{\phi_{Nh}}$ as:

$$\sigma_{\phi_{Nh}} = \frac{\pi}{2BW_{\phi_h}}\sigma_0, \tag{6.13}$$

where $BW_{\phi_h} = BW_{J_h-1}$ is the double-sided bandwidth of the low pass-filter, and $\sigma_0 = 2/\sqrt{3}$ [5].

[3] Note that the single-sided frequency band of the low-pass filter is half the frequency band of the band-pass filter $\psi_{J-1}[n]$. When required, the band of the low-pass filter can be extended, yielding a related shorter time window.

[4] Morlet and Gabor wavelets are both analytic wavelets obtained from a Gaussian window. The Morlet wavelets can be obtained from the Gabor and have zero means in the time domain. [291]

[5] The Gaussian window $\Theta[k]$ is not defined on a compact support. However, if it is assumed as a function of the discrete variable $2\pi k/N$ and defined as:

$$\Theta[k] = e^{-\frac{\sigma_0^2 2\pi k/N}{2}}, \tag{6.14}$$

with $\sigma_0 = 2/\sqrt{3}$, its amplitude can be neglected outside the interval $[-\pi, \pi]$ ($k < -N/2$, $k > N/2$). With the definition above, the (double-sided) $-3dB$ bandwidth is roughly $\pi/2$.

Along with the low-pass filter, the definition of the scattering layer requires the creation of the filter bank. As mentioned in section 3.2.3.3, J_h $\mathcal{Q}$-constant filters are generated at different scales of the mother wavelet, while the remaining lower frequencies are covered with a set of equal P_h filters spaced with constant frequency steps. Although the max index J_h depends on both B_h and Q_h, the frequency positions $\{\xi_{\psi_{h,:}}\}$ of the J_h filters depends only on Q_h, being:

$$\xi_{\psi_{h,j}} = \xi_{\psi_{h,0}} 2^{-j/Q_h} \quad with \quad \xi_{\psi_{h,0}} = \frac{1}{2}(2^{-1/Q_h}+1)\pi \quad and \quad j \in [0 : J_h - 1]. \tag{6.15}$$

When creating the filter bank, Scatnet imposes the analytic wavelets by setting the Gaussian functions only on one side of the spectrum. Nevertheless, the analytic hypothesis is violated if the non-compact support of the Gaussian function is considered. Furthermore, if the filters are excessively wide, some of them, both at low and high frequencies, may extend into the second half of the spectrum, violating the analytic hypothesis again. At low frequencies, Scatnet limits the issue by controlling the number and the position of the linearly spaced filters. At high frequencies, however, some filters may extend their support to the second half of the spectrum, violating the analytic hypothesis and introducing noisy scattering coefficients. For instance, even not considering the tail of the Gaussian beyond its bandwidth, the normalised upper boundary of filter the $\psi_{h,0}$ (the upper boundary of the filter bank h) is:

$$\psi^{+}_{h,0} = \frac{1}{2}(2^{-1/Q_h}+1)\pi + \frac{1}{2}(1-2^{-1/B_h})\pi, \tag{6.16}$$

which is higher than π for $B_h < Q_h$. To avoid the issue, an appropriate margin can be imposed between the Nyquist frequency π and the upper boundary of the inspected spectrum f_{max}. By accounting for the length of the tails, the margin can be found from:

$$f_{Nyquist} \geq f_{max} + 2BW_{\bar{j}_h} \quad with \quad \bar{j}_h : \xi_{\psi_{h,\bar{j}_h}} < f_{max}, \quad h \in [1 : H], \tag{6.17}$$

that is, as twice the bandwidth of $\psi_{h,\bar{j}_h}$, the last filter with center frequency $\xi_{\psi_{h,\bar{j}}}$ below f_{max}. Using this condition, all the scattering coefficients generated from the filters $\psi_{h,j}$ with $j < \bar{j}_h$ can be discarded.

Figures 6.20 illustrate the filters of the first and of the second layer of the scattering network used to obtain the results reported at the end of this section. The black dashed line indicates the Nyquist frequency of the sampled signal ($24kHz$), while the green dashed line indicates the upper boundary of the inspected spectrum ($12kHz$). As shown, removing the filters indicated in red avoids including filters with bands extending into the second half of the spectrum. On the contrary, choosing $f_{Nyquist}$ equal to f_{max} would return filters with part of their band in the second half of the spectrum. Note that eliminating the coefficients associated with the red filters does not discard signal information but avoids the introduction of noisy features.

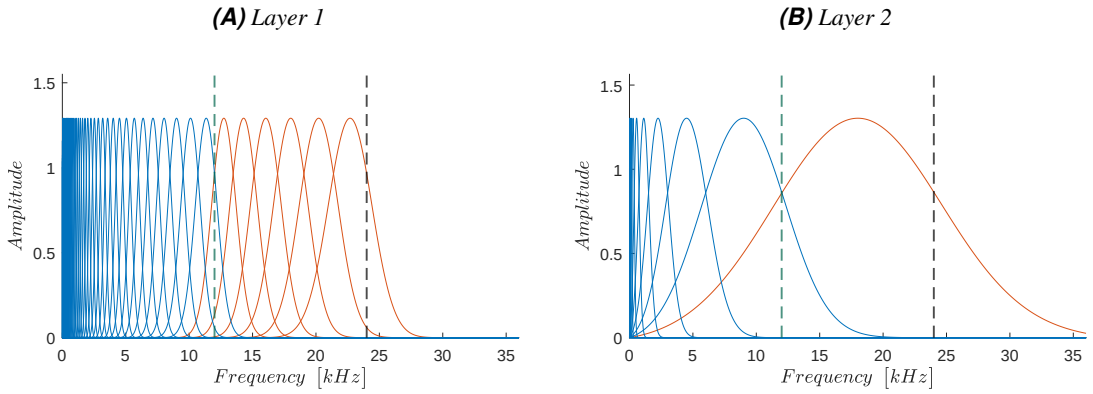


Figure 6.20: *Filter banks of the scattering network. The black dashed line indicates the Nyquist frequency. The upper boundary of the inspected spectrum is indicated by the green dashed line. The coefficients associated with the red filters are discarded since they carry no information and are potentially noisy. All the blue filters are contained in the first half of the spectrum.*

When creating the scattering network, the number of scattering layers is certainly a key parameter of the architecture. Scatnet [291] suggests that, for audio signals, using two layers is the optimal choice. Indeed the second layer suffices to recover almost all the information discarded by the first, and a third layer would disproportionately increase the complexity without adding relevant performance benefits. This setup is also the most adequate for this work, being the performance

achievable with three layers even worse as a result of the increased dimensionality and the lack of additional information retained.

Finally, the definition of the two layers requires the values for B_h and Q_h. A first approximation for Q_1 can be obtained directly from the results of the MFCC. Recalling that MFCC filters are $\mathcal{Q}$-constant at high frequencies, the equivalent number of filters per octave Q_{MFCC} can be calculated as:

$$Q_{MFCC} = \frac{\bar{f}_{max} - \bar{f}_{max/2}}{\bar{f}_{max}} F, \tag{6.18}$$

where F is the optimised number of MFCC filters, $\bar{f}_{max}$ the mel value of the linear upper boundary of the inspected spectrum f_{max}, and $\bar{f}_{max/2}$ the mel value of $f_{max}/2$. Assuming $f_{max} = 12kHz$, $F = 60$, and the corner frequency $f_0 = 115Hz$, 6.18 yields $Q_{MFCC} = 8.8$, which is close to 8, the Scatnet default value for audio processing. This value, however, does not return the optimal performance, and further optimisation is required. Therefore, the quest of the optimal Q_1 is performed by pattern search [84] using the classification edge (equation 6.6) as cost function and Q_{MFCC} as starting point. A similar operation is also applied for B_1. Both Q_1 and B_1 are optimised assuming $Q_2 = 1$ and $B_2 = 1$ as for Scatnet default. Using the collection of examples C_{A5} (table 6.2), the optimisation algorithm returns $Q_1 = 6$ and $B_1 = 5$. Values for the second layers are left as their default since further optimisation on the second layer, with Q_1 and B_1 as above, returns the same default values.

6.2.4.2 Detailed performance of the WST

Applying the settings to the scattering network as above, the resulting averaged performance over collections $C_{B0}, ..., C_{B3}$ is reported in table 6.11. The WST outperforms the STFT and the MFCC returning an improvement of the SVM F-score, over the full unitary scale, of 15.4% and 4.2%, respectively. KNN performance, despite remaining worse than SVM, shows similar or better improvements with respect to both STFT and MFCC. The confusion matrix of the classification performed over collection C_{B0} is reported in figures 6.21.

Collection	Classifier	Indicator	Average	Class					
				DW	SH	SI	TA	TO	WM
C_{B0}, C_{B1}, C_{B2}, C_{B3}	KNN	Err	0.195	0.242	0.200	0.058	0.167	0.121	0.383
		Acc	0.935	0.951	0.949	0.920	0.915	0.965	0.909
		Fsc	0.805	0.867	0.855	0.704	0.721	0.896	0.755
	SVM	Err	0.128	0.038	0.142	0.129	0.204	0.071	0.188
		Acc	0.957	0.984	0.957	0.947	0.930	0.976	0.949
		Fsc	0.872	0.952	0.873	0.837	0.788	0.927	0.853

Table 6.11: *Detailed performances of the optimised WST averaged over collections $C_{B0},...,C_{B3}$ of table 6.2. Number of scattering layers $= 2$, mother wavelet: Morlet, window length $= 500ms$, $Q_1 = 6$, $B_1 = 5$, $Q_2 = 1$, $B_2 = 1$, $f_{max} = 12kHz$, $f_{Nyquist} = 24kHz$. Total training pools per collection = 1692, total test pools per collection = 360.*

.

It is also interesting to compare the optimised performance against the one obtained using the Scatnet default audio settings. With reference to collection C_{B0}, the optimised SVM average F-score is 0.903, while the correspondent value for the Scatnet audio setting is 0.878. Besides, in the first case, the dimensionality of the features is 873, while in the second 1233. Note that the latter performance is obtained assuming $f_{max} = f_{Nyquist} = 24kHz$. The negative effect of noisy features generated by filters extending into the second half of the spectrum can be further highlighted. Indeed, resampling the signals imposing $f_{max} = f_{Nyquist} = 12kHz$ and extracting the scattering coefficients (using the optimised filters) produces the same number of features as for the optimised solution, but the SVM F-score performance returns 0.883.

Another important point concerns the efficiency of the representation. Obviously, the WST returns better results than the MFCC, but the improvement is obtained at the cost of a much higher number of features. This is easily explained by referring to the architecture of the scattering network, where the recovered information is associated with the second layer. Indeed, if WST features are compared to MFCC features imposing the same dimensionality using linear PCA, the benefit of using the WST appears less relevant. For instance, retaining only 55 principal

components and averaging over collections $C_{B0}, ..., C_{B3}$ as for table 6.11, the SVM F-score returns 0.847, a value much closer to the correspondent one for the MFCC reported in table 6.10. Interestingly, the reduced dimensionality improves the performance of the KNN classifier, which scores 0.827, a higher value compared to 0.805 of table 6.11 and 0.697 of table 6.10.

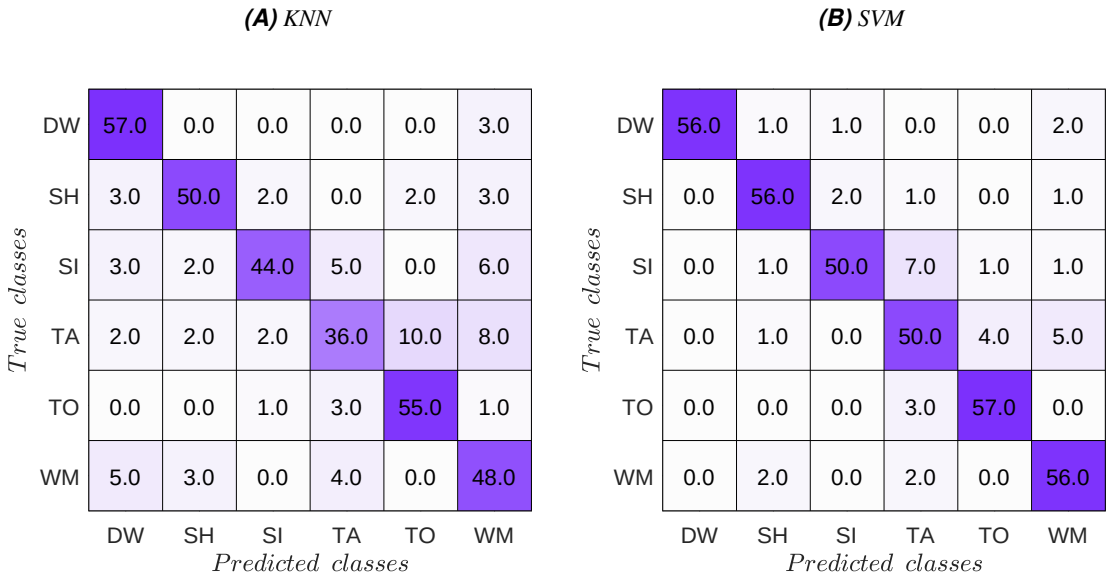


Figure 6.21: *Classification confusion matrix of the optimised WST for the collection C_{B0} of table 6.2. Number of scattering layers $= 2$, mother wavelet: Morlet, window length $= 500ms$, $Q_1 = 6$, $B_1 = 5$, $Q_2 = 1$, $B_2 = 1$, $f_{max} = 12kHz$, $f_{Nyquist} = 24kHz$. Total training pools = 1692, total test pools = 360. A) Cosine 19-NN classifier. B) Linear SVM classifier.*

6.3 Single versus multiple folds

In the previous sections, the collections of examples employed are obtained using the single-fold approach defined in section 6.1. Although the same source audio files cannot be shared between the training and the test set, the single-fold approach allows different recordings of the same acoustic source to be divided between the test and training sets. This section briefly describes the performance of a collection of examples obtained using the multiple-fold approach, that is, strictly separating the acoustic sources between the test and the training set. This condition is clearly more restrictive and less tolerant to the limited intra-class variability in the soundbank. The multi-fold collection of examples used is C_{C0} (table 6.2).

Collection	Classifier	Indicator	Average	Class					
				DW	SH	SI	TA	TO	WM
C_{B0}, C_{B1}, C_{B2}, C_{B3}	KNN	Err	0.439	0.606	0.562	0.170	0.548	0.192	0.554
		Acc	0.855	0.886	0.851	0.857	0.794	0.912	0.837
		Fsc	0.566	0.732	0.594	0.411	0.325	0.699	0.518
	SVM	Err	0.335	0.132	0.398	0.327	0.688	0.223	0.244
		Acc	0.889	0.953	0.873	0.902	0.792	0.916	0.900
		Fsc	0.669	0.849	0.618	0.716	0.410	0.726	0.677

Table 6.12: *Detailed performances of the WST for the 4-folds collection C_{C0}. Minimum training pools per fold = 2340, minimum test pools per fold = 774. Features are extracted with the same procedure used for table 6.11.*

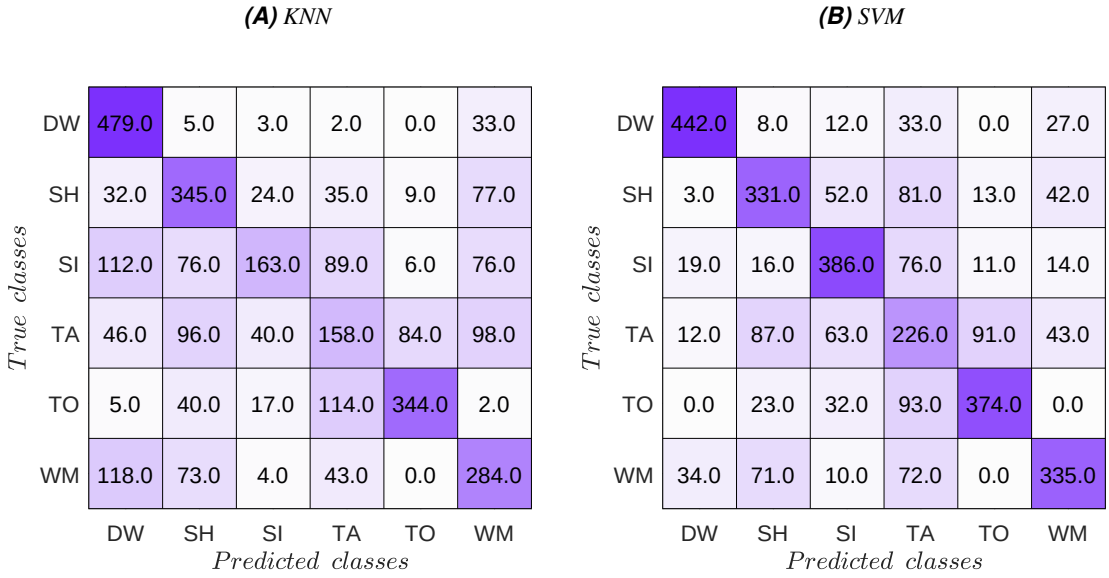


Figure 6.22: *Classification confusion matrix of the WST for the 4-folds collection C_{C0}. Minimum training pools per fold = 2340, minimum test pools per fold = 774. Features are extracted as from table 6.11. A) Cosine 19-NN classifier. B) Linear SVM classifier.*

Table 6.12 and figures 6.22 report, respectively, the detailed performance and the classification confusion matrix for both SVM and KNN. The features are extracted using WST as for table 6.11. Although KNN results can be improved by $4 \div 5\%$ using linear PCA, the overall performance is lower for both classifiers. Interestingly, the reduction of the performance is not uniform, with some classes, such as *Tap*,

more affected and others, such as *Dish Washers* and *Toilets*, closer to the single fold results. The class *Tap* is also the worse performing in the single-fold case, suggesting that stronger intra-class variability is necessary in the soundbank. In general, however, more comprehensive intra-class variability is required for all the classes. Table 6.1 shows that the number of sources used to build the dataset is quite limited and, despite the benefit provided by the artificial synthesis, the acquisition of a good number of real sources remains crucial. Indeed, since the collection of sounds used are noise-like and (acoustically) poorly structured, wider coverage of the in-class variability is necessary, especially if classifiers are supposed to analyse unknown sound sources.

6.4 Summary

This chapter shows how the created dataset can be processed to extract optimised features suitable for machine learning applications. A particular synthesis strategy, the single instance foldable synthesis, is proposed to guide the synthesis of the dataset and produce observations that can be conveniently partitioned into training and test sets, keeping either source observations or source instances strictly separate. Three sets of example collections with different properties are extracted from single and multi-fold partitions to conduct performance tests. A minimum number of 1692 pools per collection (2340 for the multi-fold) are used for the training sets, while 360 pools per collection (774 for the multi-fold) are used for the test sets. Performance tests focus on three different signal representations: short-time Fourier transform, mel-frequency cepstral coefficients, and wavelet scattering transform. The processing chain adopted employs K-nearest neighbours and support vector machines as classifiers. A range of different parameters is optimised to tailor the chosen signal representations, increasing the efficiency and the performance of the extracted features. For instance, the retained spectrum is reduced to $0-12kHz$, and the optimal length of the windows ($500ms$) is found to be longer than the one usually adopted for machine learning audio applications ($20 \div 50ms$). The Gauss windowing function is conveniently adopted as a low-pass filter for windows of such a length. The importance of non-linear feature scaling,

as for bio-inspired feature solutions, is confirmed by several percentage points of performance improvement, and it is obtainable either at the feature or at the classifier level. Dimensionality reduction is preferably obtained through linear PCA since no relevant benefit has been found with non-linear kernel solutions. MFCC and WST are identified as viable options for obtaining light representations or boosting performance, respectively. To optimise the MFCC, a novel algorithm is proposed to find the optimal number of filters and a more convenient mel-Hz frequency conversion law. The optimised features, which are more detailed at low frequencies, exhibit average F-score performance roughly 6% higher. The number of filters and the related bands of the WST are also optimised using a similar algorithm. An additional truncation of the filter banks is adopted to mitigate the effect of noisy features related to the violation of the analytical hypothesis of the wavelets. The optimised WST yields a 4.2% F-score improvement compared to the optimised MFCC, and a 2.5% improvement compared to the default Scatnet configuration. The contribution of the filter truncation to the latter is around 2% out of the total 2.5%. Overall, the results show that the size of the acquired soundbank is reasonable only if the machine is supposed to process known sound sources, while more comprehensive coverage of the intra-class variability is necessary for other applications. Additional machine learning tasks, such as event detection, multi-class classification, and disturbances discrimination, can be developed as future extensions of this chapter. The analysis of the representation robustness as a function of reverberation phenomena of different magnitudes is also left for future developments.

Chapter 7

Conclusions

In this chapter, the main conclusions are summarised along with the related limitations. A further overview of possible industrial applications and future developments is provided.

7.1 Conclusions, remarks, limitations

This work proposes a development chain for in-pipe acoustic machine-learning applications. Starting from a comprehensive review of the existing literature, it is demonstrated how the usage of a relatively limited amount of real data can be optimised to create one or more datasets for the implementation of machine learning algorithms. Different signal representations and transformations are investigated and analysed to extract suitable solutions for the acoustic features. A baseline classification performance is obtained as a benchmark for further research.

The creation of the soundbank is the first crucial node of the processing chain. It is shown how the human labour required to generate a new dataset can be minimised by avoiding the manual creation of strong annotations. The only manual collection of metadata required is reduced to weak annotations obtained by sorting the audio files by class instances in different folders. Strong, rich, and precise labels are created automatically during the synthesis and stored in the machine and human-readable format `.jams`.

The post-processing of the soundbank removes the background noise from the source recordings using a statistical spectral noise filter. A novel solution is proposed and evaluated against close references in the literature. Filtering the background noise from the source recordings allows precise control of the signal-to-noise ratio in the synthetic observations and a more accurate determination of the event on-set and off-set.

Although the number of source recordings remains crucial for the synthesised dataset, several artifices are implemented to obtain increased intra-class variability for a given soundbank. For instance, duration-constant pitch shift and pitch-constant dilation are applied on top of artificial events obtained by recombining distinct source observations belonging to the same instances. Care is taken to promote natural-like results. Acoustic properties of the generated synthetic observations, such as signal-to-noise ratio, duration of the events, and the number of overlapping events, can be finely controlled by assigning the boundaries of the related random variation ranges.

Synthetic observations are also reverberated using the acoustic model developed. The model is implemented using a modal decomposition analysis and simulates the dispersive effects introduced by the propagation of sounds in linear elastic waveguides. Novel solutions are proposed for mode extraction and separation, along with a procedure to model cross-sectional pressure sources. Using a mixture of numerical and analytical solutions and an implementation on three different layers, the model provides calculation time much lower than what is achievable using finite element software. Although this advantage also comes with several limitations (e.g. the options for the allowed geometries are limited), the model remains flexible enough to be matched to simple real in-house test rigs. A further limitation, certainly surmountable in future releases, is given by the unaccounted modal dumping. As a consequence, the relative magnitude of the propagating modes is unrealistically assumed constant along the waveguide.

An arbitrary number of pseudo-random synthetic observations can be generated and organised in synthetic datasets. The synthesis is guided using a procedure that has been called single-instance foldable synthesis. This procedure allows the

creation of improved balanced dataset partitions despite the possible imbalance in the soundbank. The single or multi-fold partitions generated are truly non-overlapping, meaning that test and training partitions cannot contain synthetic observations with source audio files or source audio instances in common. This is also achievable since the synthesis maintains its entire history clearly recorded in the links between the soundbank and the dataset in the annotations. It is remarked that single and multi-fold partitions use two different separation strategies. While the single-fold allows different observations belonging to the same instances in both test and training partitions, the multi-fold imposes strict separation of the instances. The latter means that the same physical source cannot be associated with both test and training partitions in the same fold.

The obtained partitions are used to perform machine-learning tasks. In particular, representations based on short-time Fourier transform, mel-frequency cepstral coefficients, and wavelet scattering transform are optimised with respect to classification tasks using K-nearest neighbours and support vector machine classifiers. The importance of very low-frequency components is shown. Window duration, window shape, filter banks, and feature dimensionality are among the optimised quantities. A novel algorithm is proposed to optimise the number of filters and the mel-Hz conversion law for the MFCC. A similar procedure is also applied for the WST, where a truncation of the filter bank is also introduced to discard noisy features. PCA and kernel PCA are applied to investigate the reduction and non-linear transformation of the features. The efficiency of the representation as performance versus dimensionality is also discussed. Single-fold and multi-fold results are compared, showing that a wider inter-class representation in the soundbank is necessary when unknown sound sources need analysing. Finally, a set of results is provided as a benchmark for future developments.

7.2 Future developments

The results obtained in this research can be applied for the development of real in-pipe event detectors. Better results, both in terms of the accuracy and the variety of events detectable, can be achieved by combining acoustic solutions with data

provided by other sensors, such as flow, temperature, and turbidity. An example of this integration is given by the multi-purpose measurement unit developed by Aquacheck Engineering Ltd (figure 7.1), which is designed to support an acoustic detector along with other sensing devices under the same physical case.

Further research is undoubtedly required to gain a better insight into other machine learning tasks such as event detection and multi-class/disturbance discrimination. These investigations can be conducted on the same datasets as a direct extension of this work. From a broader perspective, additional research is undoubtedly required to blend heterogeneous data, and the approach developed in this work can be extended and complemented. Nevertheless, acquiring and collecting representative sets of examples in soundbanks (or, more generically, in sample banks) remains challenging. Mathematical models can certainly help by simulating data that cannot be easily acquired. Indeed, further improvements for audio data can be obtained by refining the reverberation model and accounting for modal dumping and wave scattering.

Figure 7.1: *Aquacheck Engineering multi-purpose measurement unit.*

References

[1] A. G. Yilmaz, I. Hossain, and B. J. C. Perera. “Effect of climate change and variability on extreme rainfall intensity–frequency–duration relationships: a case study of Melbourne”. In: *Hydrology and Earth System Sciences* 18.10 (2014), pp. 4065–4076. DOI: 10.5194/hess-18-4065-2014. URL: https://hess.copernicus.org/articles/18/4065/2014/.

[2] S. Javadinejad, R. Dara, and F. Jafary. “Climate Change Scenarios and Effects on Snow-Melt Runoff”. In: *Civil Engineering Journal* 6 (Sept. 2020), pp. 1715–1725. DOI: 10.28991/cej-2020-03091577.

[3] C. Notarnicola. “Overall negative trends for snow cover extent and duration in global mountain regions over 1982–2020”. In: *Scientific Reports* 12 (Aug. 2022). DOI: 10.1038/s41598-022-16743-w.

[4] J. C. Hammond, F. A. Saavedra, and S. K. Kampf. “Global snow zone maps and trends in snow persistence 2001–2016”. In: *International Journal of Climatology* 38.12 (2018), pp. 4369–4383. DOI: https://doi.org/10.1002/joc.5674. URL: https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/joc.5674.

[5] H. Tabari. “Climate change impact on flood and extreme precipitation increases with water availability”. In: *Scientific Reports* 10 (Aug. 2020), p. 13768. DOI: 10.1038/s41598-020-70816-2.

[6] S. Hettiarachchi, C. Wasko, and A. Sharma. “Increase in flood risk resulting from climate change in a developed urban watershed - the role of storm temporal patterns”. In: *Hydrology and Earth System Sciences*

22.3 (2018), pp. 2041–2056. DOI: 10.5194/hess-22-2041-2018. URL: https://hess.copernicus.org/articles/22/2041/2018/.

[7] N. Gaaloul, E. Saeid, and K. Rim. “Impacts of Climate Change and Water Resources Management in the Southern Mediterranean Countries”. In: V (Nov. 2020).

[8] P. A. Dirmeyer et al. “Land-Atmosphere Interactions Exacerbated the Drought and Heatwave Over Northern Europe During Summer 2018”. In: *AGU Advances* 2.2 (2021). e2020AV000283 2020AV000283, e2020AV000283. DOI: https://doi.org/10.1029/2020AV000283. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2020AV000283. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020AV000283.

[9] J. Mateo-Sagasta, S. Marjani, and H. Turral. *More people, more food, worse water?: A global review of water pollution from agriculture*. FAO, Aug. 2018. ISBN: 978-92-5-130729-8.

[10] P. D’Odorico et al. “The global value of water in agriculture”. In: *Proceedings of the National Academy of Sciences* 117.36 (2020), pp. 21985–21993. ISSN: 0027-8424. DOI: 10.1073/pnas.2005835117. eprint: https://www.pnas.org/content/117/36/21985.full.pdf. URL: https://www.pnas.org/content/117/36/21985.

[11] I. N. Daliakopoulos et al. “The threat of soil salinity: A European scale review”. In: *Science of The Total Environment* 573 (2016), pp. 727–739. ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2016.08.177. URL: https://www.sciencedirect.com/science/article/pii/S0048969716318794.

[12] J. Rodriguez et al. “Food production link to underground waters quality in A Limia river basin”. In: *Agriculture, Ecosystems Environment* 297 (2020), p. 106969. ISSN: 0167-8809. DOI: https://doi.org/10.1016/j.agee.2020.106969. URL: https://www.sciencedirect.com/science/article/pii/S0167880920301547.

[13] N. Omrani. “Dilemma of fossil water management within Southern Tunisia oases: vulnerability to salt under intensive use context”. In: *8th World Wide Workshop for Young Environmental Scientists WWW-YES 2009: Urban waters: resource or risks?* Ed. by D. Thevenot. Vol. WWW-YES-2009-Fr. WWW-YES 8. June 2009. URL: https://hal.archives-ouvertes.fr/hal-00591658.

[14] P. D’Odorico et al. “The Global Food-Energy-Water Nexus”. In: *Reviews of Geophysics* 56.3 (2018), pp. 456–531. DOI: https://doi.org/10.1029/2017RG000591. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2017RG000591. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2017RG000591.

[15] M. Flörke, C. Schneider, and R. Mcdonald. “Water competition between cities and agriculture driven by climate change and urban growth”. In: *Nature Sustainability* 1 (Jan. 2018). DOI: 10.1038/s41893-017-0006-8.

[16] M. Hameed et al. “A Review of the 21st Century Challenges in the Food-Energy-Water Security in the Middle East”. In: *Water* 11.4 (2019). ISSN: 2073-4441. DOI: 10.3390/w11040682. URL: https://www.mdpi.com/2073-4441/11/4/682.

[17] M. Arfanuzzaman and A. Atiq Rahman. “Sustainable water demand management in the face of rapid urbanization and ground water depletion for social–ecological resilience building”. In: *Global Ecology and Conservation* 10 (2017), pp. 9–22. ISSN: 2351-9894. DOI: https://doi.org/10.1016/j.gecco.2017.01.005. URL: https://www.sciencedirect.com/science/article/pii/S235198941630141X.

[18] E. Jones et al. “The state of desalination and brine production: A global outlook”. In: *Science of The Total Environment* 657 (2019), pp. 1343–1356. ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2018.12.076. URL: https://www.sciencedirect.com/science/article/pii/S0048969718349167.

[19] M. Stavenhagen, J. Buurman, and C. Tortajada. “Saving water in cities: Assessing policies for residential water demand management in four cities in Europe”. In: *Cities* 79 (2018), pp. 187–195. ISSN: 0264-2751. DOI: https://doi.org/10.1016/j.cities.2018.03.008. URL: https://www.sciencedirect.com/science/article/pii/S0264275117307655.

[20] D. B. Brooks and O. M. Brandes. “Why a Water Soft Path, Why Now and What Then?” In: *International Journal of Water Resources Development* 27.2 (2011), pp. 315–344. DOI: 10.1080/07900627.2011.571235. URL: https://doi.org/10.1080/07900627.2011.571235.

[21] A. Awwad et al. “Communication Network for Ultrasonic Acoustic Water Leakage Detectors”. In: *IEEE Access* 8 (2020), pp. 29954–29964. DOI: 10.1109/ACCESS.2020.2972648.

[22] B. Lu et al. “Recent Advances of Hyperspectral Imaging Technology and Applications in Agriculture”. In: *Remote Sensing* 12.16 (2020). ISSN: 2072-4292. DOI: 10.3390/rs12162659. URL: https://www.mdpi.com/2072-4292/12/16/2659.

[23] H. March et al. “Household Smart Water Metering in Spain: Insights from the Experience of Remote Meter Reading in Alicante”. In: *Sustainability* 9.4 (2017). ISSN: 2071-1050. DOI: 10.3390/su9040582. URL: https://www.mdpi.com/2071-1050/9/4/582.

[24] S. Eggimann et al. “The Potential of Knowing More: A Review of Data-Driven Urban Water Management”. In: *Environmental Science & Technology* 51.5 (2017). PMID: 28125222, pp. 2538–2553. DOI: 10.1021/acs.est.6b04267. eprint: https://doi.org/10.1021/acs.est.6b04267. URL: https://doi.org/10.1021/acs.est.6b04267.

[25] A. Cominola et al. “Benefits and challenges of using smart meters for advancing residential water demand modeling and management: A review”. In: *Environmental Modelling Software* 72 (2015), pp. 198–214. ISSN: 1364-8152. DOI: https://doi.org/10.1016/j.envsoft.2015.07.012. URL: https://www.sciencedirect.com/science/article/pii/S1364815215300177.

[26] S. Behmel et al. “Water quality monitoring strategies — A review and future perspectives”. In: *Science of The Total Environment* 571 (2016), pp. 1312–1329. ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2016.06.235. URL: https://www.sciencedirect.com/science/article/pii/S0048969716314243.

[27] Anglian Water. *Anglian Water: Monitoring drinking water quality*. 2022. URL: https://www.anglianwater.co.uk/services/water-supply/monitoring-drinking-water-quality/.

[28] Welsh Water. *Welsh Water: providing clean water*. 2022. URL: https://corporate.dwrcymru.com/en/community/education/teaching-resources/secondary-resources/providing-clean-water.

[29] W. Robertson et al. “Monitoring the Quality of Drinking-water During Storage and Distribution”. In: Jan. 2003.

[30] F. Ghobadi, G. Jeong, and D. Kang. “Water Pipe Replacement Scheduling Based on Life Cycle Cost Assessment and Optimization Algorithm”. In: *Water* 13.5 (2021). ISSN: 2073-4441. DOI: 10.3390/w13050605. URL: https://www.mdpi.com/2073-4441/13/5/605.

[31] United Utilities. *United Utilities: discoloured water*. 2022. URL: https://www.unitedutilities.com/help-and-support/your-water-supply/your-water/water-quality/water-appearance/discoloured-water/.

[32] S. Msamadya et al. “Role of Water Policies in the Adoption of Smart Water Metering and the Future Market”. In: *Water* 14.5 (2022). ISSN: 2073-4441. DOI: 10.3390/w14050826. URL: https://www.mdpi.com/2073-4441/14/5/826.

[33] M. R. Asghar et al. “Smart Meter Data Privacy: A Survey”. In: *IEEE Communications Surveys Tutorials* 19.4 (2017), pp. 2820–2835. DOI: 10.1109/COMST.2017.2720195.

[34] S Finster and I Baumgart. “Privacy-Aware Smart Metering: A Survey”. In: *IEEE Communications Surveys Tutorials* 16.3 (2014), pp. 1732–1745. DOI: 10.1109/SURV.2014.052914.00090.

[35] Thames Water Utilities Ltd. *Thames Water: getting a standpipe*. 2022. URL: https://www.thameswater.co.uk/help/home-improvements/standpipes.

[36] P. F. Boulos et al. “Hydraulic Transient Guidelines for Protecting Water Distribution Systems”. In: *Journal AWWA* 97.5 (2005), pp. 111–124. DOI: https://doi.org/10.1002/j.1551-8833.2005.tb10892.x. URL: https://awwa.onlinelibrary.wiley.com/doi/abs/10.1002/j.1551-8833.2005.tb10892.x.

[37] Manchester Metropolitan University. *Aquacheck: A Knowledge Transfer Partnership*. 2022. URL: https://www.mmu.ac.uk/research/our-impact/case-studies/aquacheck.

[38] T. Virtanen, M. D. Plumbley, and D. Ellis. *Computational Analysis of Sound Scenes and Events*. Springer International Publishing, 2017. ISBN: 9783319634500. URL: https://books.google.co.uk/books?id=luQ2DwAAQBAJ.

[39] A. Mesaros, T. Heittola, and T. Virtanen. *TUT Sound events 2017, Development dataset*. Mar. 2017. DOI: 10.5281/zenodo.814831. URL: https://doi.org/10.5281/zenodo.814831.

[40] M. Cartwright et al. *SONYC Urban Sound Tagging (SONYC-UST): a multilabel dataset from an urban acoustic sensor network*. Version 2.3.0. Sept. 2020. DOI: 10.5281/zenodo.3966543. URL: https://doi.org/10.5281/zenodo.3966543.

[41] W. Liang et al. “Author Correction: Advances, challenges and opportunities in creating data for trustworthy AI”. In: *Nature Machine Intelligence* (Sept. 2022), pp. 1–1. DOI: 10.1038/s42256-022-00548-7.

[42] Amazon.com Inc. *Amazon Mechanical Turk*. 2022. URL: https://www.mturk.com/.

[43] CloudFactory Ltd. *CloudFactory*. 2022. URL: https://www.cloudfactory.com/.

[44] S. Hantke et al. “Trustability-Based Dynamic Active Learning for Crowdsourced Labelling of Emotional Audio Data”. In: *IEEE Access* 6 (2018), pp. 42142–42155. DOI: 10.1109/ACCESS.2018.2858931.

[45] K. Woodward et al. “LabelSens: enabling real-time sensor data labelling at the point of collection using an artificial intelligence-based approach”. In: *Personal and Ubiquitous Computing* 24 (Oct. 2020). DOI: 10.1007/s00779-020-01427-x.

[46] J. Ma et al. “High-Sensitivity Ultrasonic Guided Wave Monitoring of Pipe Defects Using Adaptive Principal Component Analysis”. In: *Sensors* 21.19 (2021). ISSN: 1424-8220. DOI: 10.3390/s21196640. URL: https://www.mdpi.com/1424-8220/21/19/6640.

[47] N. Mashhadi et al. “Use of Machine Learning for Leak Detection and Localization in Water Distribution Systems”. In: *Smart Cities* 4 (Oct. 2021), pp. 1293–1314. DOI: 10.3390/smartcities4040069.

[48] H. Shukla and K. Piratla. “Leakage detection in water pipelines using supervised classification of acceleration signals”. In: *Automation in Construction* 117 (2020), p. 103256. ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2020.103256. URL: https://www.sciencedirect.com/science/article/pii/S0926580519310301.

[49] A. Rai and J. M. Kim. “A novel pipeline leak detection approach independent of prior failure information”. In: *Measurement* 167 (2021), p. 108284. ISSN: 0263-2241. DOI: https://doi.org/10.1016/j.measurement.2020.108284. URL: https://www.sciencedirect.com/science/article/pii/S0263224120308241.

[50] A. Kadri, E. Yaacoub, and M. Mushtaha. “Empirical evaluation of acoustical signals for leakage detection in underground plastic pipes”. In: *MELECON 2014 - 2014 17th IEEE Mediterranean Electrotechnical Conference*. 2014, pp. 54–58. DOI: 10.1109/MELCON.2014.6820506.

[51] S. Chen et al. “Construction and Experimental Research on Leakage Sound Dataset of Urban Water Supply Pipeline”. In: *2021 China Automation Congress (CAC)*. 2021, pp. 5385–5390. DOI: 10.1109/CAC53003.2021.9727424.

[52] T. Sainburg, M. Thielk, and T. Q. Gentner. “Finding, visualizing, and quantifying latent structure across diverse animal vocal repertoires”. In: *PLOS Computational Biology* 16.10 (Oct. 2020), pp. 1–48. DOI: 10.1371/journal.pcbi.1008228. URL: https://doi.org/10.1371/journal.pcbi.1008228.

[53] M. H. Hayes. *Statistical Digital Signal Processing and Modeling*. John Wiley and Sons. Inc., 1996. ISBN: 978-0-471-59431-4.

[54] A. Copiaco et al. “Development of a Synthetic Database for Compact Neural Network Classification of Acoustic Scenes in Dementia Care Environments”. In: *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. 2021, pp. 1202–1209.

[55] X. Zheng et al. “Using Synthetic Audio to Improve the Recognition of Out-of-Vocabulary Words in End-to-End Asr Systems”. In: *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021, pp. 5674–5678. DOI: 10.1109/ICASSP39728.2021.9414778.

[56] R. Serizel et al. “Sound Event Detection in Synthetic Domestic Environments”. In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 86–90. DOI: 10.1109/ICASSP40776.2020.9054478.

[57] K. Kinoshita et al. “A Summary of the REVERB Challenge: State-of-the-Art and Remaining Challenges in Reverberant Speech Processing Research”. In: *Journal on Advances in Signal Processing* 2016 (Jan. 2016). DOI: 10.1186/s13634-016-0306-6.

[58] J. Salamon et al. “Scaper: A Library for Soundscape Synthesis and Augmentation”. In: Oct. 2017. DOI: 10.1109/WASPAA.2017.8170052.

[59] T. Wendt, S. Van De Par, and S.D. Ewert. “A computationally-efficient and perceptually-plausible algorithm for binaural room impulse response simulation”. In: *journal of the audio engineering society* 62.11 (Nov. 2014), pp. 748–766. DOI: https://doi.org/10.17743/jaes.2014.0042.

[60] C. Schissler and D. Manocha. “GSound: Interactive Sound Propagation for Games”. In: Feb. 2011.

[61] F. Fontana and D. Rocchesso. “Auditory Distance Perception in an Acoustic Pipe”. In: *ACM Trans. Appl. Percept.* 5.3 (Sept. 2008). ISSN: 1544-3558. DOI: 10.1145/1402236.1402240. URL: https://doi.org/10.1145/1402236.1402240.

[62] C. Liu, L. Wang, and J. Dang. “Masking based Spectral Feature Enhancement for Robust Automatic Speech Recognition”. In: *2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*. 2020, pp. 287–291. DOI: 10.1109/ICAICA50127.2020.9181915.

[63] K. Chu et al. “Using machine learning to mitigate the effects of reverberation and noise in cochlear implants”. In: *Proceedings of Meetings on Acoustics* 33.1 (2018), p. 050003. DOI: 10.1121/2.0000905. URL: https://asa.scitation.org/doi/abs/10.1121/2.0000905.

[64] T. S. Såstad and K. T. Hjelmervik. “Synthesizing Realistic, High-Resolution Anti-Submarine Sonar Data”. In: *2018 OCEANS - MTS/IEEE Kobe Techno-Oceans (OTO)*. 2018, pp. 1–5. DOI: 10.1109/OCEANSKOBE.2018.8558837.

[65] E. L. Kinsler et al. *Fundamentals Of Acoustics*. Wiley, 2000. ISBN: 978-0-471-84789-2.

[66] G. Kokossalalus. “Acoustic Data Communication system for in-pipe wireless sensor networks”. PhD thesis. Massachusetts Institute of Technology, 2006.

[67] L. Jing et al. “Channel Characterization of Acoustic Waveguides Consisting of Straight Gas and Water Pipelines”. In: *IEEE Access* 6.20 (2018), pp. 6807–6819. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2793299.

[68] T. Kundu. *Mechanics of Elastic Waves and Ultrasonic Nondestructive Evaluation*. CRC Press, 2019.

[69] V. Valimaki et al. “Fifty Years of Artificial Reverberation”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 20.5 (2012), pp. 1421–1448. DOI: 10.1109/TASL.2012.2189567.

[70] M. Holters, T. Corbach, and U. Zölzer. “Impulse Response Measurement Techniques and their Applicability in the Real World”. In: Sept. 2009.

[71] Z. Tang et al. “GWA: A Large High-Quality Acoustic Dataset for Audio Processing”. In: *ACM SIGGRAPH 2022 Conference Proceedings*. SIGGRAPH ’22. Vancouver, BC, Canada: Association for Computing Machinery, 2022. ISBN: 9781450393379. DOI: 10.1145/3528233.3530731. URL: https://doi.org/10.1145/3528233.3530731.

[72] V. Meleshko et al. “Elastic waveguides: History and the state of the art. I”. In: *Journal of Mathematical Sciences* 162.10 (Oct. 2009), pp. 99–120. DOI: 10.1007/s10958-009-9623-8.

[73] K. F. Graff. *Wave Motion in Elastic Solids*. Dover Publications, 1991. ISBN: 0-486-66745-6.

[74] M. H. Sadd. *Elasticity - Theory, Applications, and Numerics*. Academic Press, 2009. ISBN: 978-0-12-374446-3.

[75] S. Moore. “A review of noise and vibration in fluid-filled pipe systems”. In: 2016, pp. 9–11.

[76] K. Baik, J. Jiang, and T. G. Leighton. “Acoustic attenuation, phase and group velocities in liquid-filled pipes III: Nonaxisymmetric propagation and circumferential modes in lossless conditions”. In: *The Journal of the Acoustical Society of America* 133.3 (2013), pp. 1225–1236. DOI: 10.1121/1.4773863.

[77] M. Moreaux et al. “Benchmark for Kitchen20, a daily life dataset for audio-based human action recognition”. In: *2019 International Conference on Content-Based Multimedia Indexing (CBMI)*. 2019, pp. 1–6. DOI: 10.1109/CBMI.2019.8877429.

[78] N. Turpault et al. “Sound Event Detection and Separation: A Benchmark on Desed Synthetic Soundscapes”. In: *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021, pp. 840–844. DOI: 10.1109/ICASSP39728.2021.9414789.

[79] J. Andén and S. Mallat. “Deep Scattering Spectrum”. In: *IEEE Transactions on Signal Processing* 62 (Apr. 2013). DOI: 10.1109/TSP.2014.2326991.

[80] Mallat S. *A Wavelet Tour of Signal Processing (Third Edition)*. Ed. by Mallat S. Third Edition. Boston: Academic Press, 2009. ISBN: 978-0-12-374370-1. DOI: https://doi.org/10.1016/B978-0-12-374370-1.00008-2. URL: https://www.sciencedirect.com/science/article/pii/B9780123743701000082.

[81] T. Giannakopoulos and A. Pikrakis. “Chapter 4 - Audio Features”. In: *Introduction to Audio Analysis*. Ed. by T. Giannakopoulos and A. Pikrakis. Oxford: Academic Press, 2014, pp. 59–103. ISBN: 978-0-08-099388-1. DOI: https://doi.org/10.1016/B978-0-08-099388-1.00004-2. URL: https://www.sciencedirect.com/science/article/pii/B9780080993881000042.

[82] S. Mallat. “Group Invariant Scattering”. In: *Communications on Pure and Applied Mathematics* 65.10 (2012), pp. 1331–1398. DOI: https://doi.org/10.1002/cpa.21413. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.21413. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.21413.

[83] F. Zheng, G. Zhang, and Z. Song. “Comparison of Different Implementations of MFCC”. In: *J. Comput. Sci. Technol.* 16 (Nov. 2001), pp. 582–589. DOI: 10.1007/BF02943243.

[84] C. Audet and J. E. Dennis. “Analysis of Generalized Pattern Searches”. In: *SIAM Journal on Optimization* 13.3 (2002), pp. 889–903. DOI: 10.1137/S1052623400378742.

[85] R. A. Rutenbar. “Simulated annealing algorithms: an overview”. In: *IEEE Circuits and Devices Magazine* 5.1 (1989), pp. 19–26. DOI: 10.1109/101.17235.

[86] C. Cortes and V. Vapnik. “Support-vector networks”. In: *Chem. Biol. Drug Des.* 297 (Jan. 2009), pp. 273–297. DOI: 10.1007/%2FBF00994018.

[87] Mathwork Inc. *Matlab: resubedge*. 2021. URL: https://uk.mathworks.com/help/stats/classificationecoc.resubedge.html#buh1ow3-1_sep_shared-Escalera2009.

[88] B. A. Auld. *Acoustic Fields and Waves in Solids*. v. 2. Wiley, 1973. ISBN: 0-471-03701-x. URL: https://books.google.co.uk/books?id=T56ptQEACAAJ.

[89] K. Baik, J. Jiang, and T. G. Leighton. “Acoustic attenuation, phase and group velocities in liquid-filled pipes: Theory, experiment, and examples of water and mercury”. In: *The Journal of the Acoustical Society of America* 128.5 (2010), pp. 2610–2624. DOI: 10.1121/1.3495943.

[90] O. Mesnil et al. “Simulation tools for guided wave based structural health monitoring”. In: *AIP Conference Proceedings* 1949.1 (2018), p. 050001. DOI: 10.1063/1.5031543. eprint: https://aip.scitation.org/doi/pdf/10.1063/1.5031543. URL: https://aip.scitation.org/doi/abs/10.1063/1.5031543.

[91] M. Abbas and M. Shafiee. “Structural Health Monitoring (SHM) and Determination of Surface Defects in Large Metallic Structures using Ultrasonic Guided Waves”. In: *Sensors* 18.11 (2018). ISSN: 1424-8220. DOI: 10.3390/s18113958. URL: https://www.mdpi.com/1424-8220/18/11/3958.

[92] J. C. Olivier. *Linear Systems and Signals: A Primer*. Artech House radar library. Artech House, 2018. ISBN: 9781630816155. URL: https://books.google.co.uk/books?id=XfCDDwAAQBAJ.

[93] A. Southern et al. “Room Impulse Response Synthesis and Validation Using a Hybrid Acoustic Model”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 21.9 (2013), pp. 1940–1952. DOI: 10.1109/TASL.2013.2263139.

[94] P. Guidorzi et al. “Impulse Responses Measured with MLS or Swept-Sine Signals Applied to Architectural Acoustics: An In-depth Analysis of the Two Methods and Some Case Studies of Measurements Inside Theaters”. In: vol. 78. 6th International Building Physics Conference, IBPC 2015. 2015, pp. 1611–1616. DOI: https://doi.org/10.1016/j.egypro.2015.11.236. URL: https://www.sciencedirect.com/science/article/pii/S1876610215019682.

[95] M. R. Schroeder. “Integrated-impulse method measuring sound decay without using impulses”. In: *The Journal of the Acoustical Society of America* 66.2 (1979), pp. 497–500. DOI: 10.1121/1.383103. eprint: https://doi.org/10.1121/1.383103. URL: https://doi.org/10.1121/1.383103.

[96] A. Farina. “Advancements in impulse response measurements by sine sweeps”. In: 2007.

[97] G. B. Stan, J. J. Embrechts, and D. Archambeau. “Comparison of different impulse response measurement techniques”. In: *Journal of the Audio Engineering Society* 50 (Apr. 2002), pp. 249–262.

[98] M. Mekarzia and M. Guerti. “Measurement and Identification of Acoustic Impulse Response”. In: *Building Acoustics* 15.1 (2008), pp. 73–78. DOI: 10.1260/135101008784050197. eprint: https://doi.org/10.1260/135101008784050197. URL: https://doi.org/10.1260/135101008784050197.

[99] L. Jing, Y. Li, and R. D. Murch. “Wideband modeling of the acoustic water pipe channel”. In: *OCEANS 2016 - Shanghai*. 2016, pp. 1–8. DOI: 10.1109/OCEANSAP.2016.7485640.

[100] P. Drábek and G. Holubová. *Elements of Partial Differential Equations*. Berlin, Boston: De Gruyter, 2014. ISBN: 9783110316674. DOI: doi:10.1515/9783110316674. URL: https://doi.org/10.1515/9783110316674.

[101] D. R. Bergman. *Computational acoustics: theory and implementation*. John Wiley & Sons, 2018. ISBN: 978-1-119-27728-6.

[102] P. J. Olver. *Introduction to Partial Differential Equations*. Springer International Publishing, 2014. ISBN: 978-3-319-02098-3. DOI: 10.1007/978-3-319-02099-0.

[103] J. D. Logan. “Applied Partial Differential Equations”. In: Springer New York, NY, 2004. ISBN: 978-0-387-20953-1. DOI: https://doi.org/10.1007/978-1-4419-8879-9.

[104] T. Hillen, I. Leonard, and H. van Roessel. *Partial Differential Equations, Theory and Completely Solved Problems*. Wiley, Jan. 2012. ISBN: 978-1-118-06330-9.

[105] J. Alonso and R. Burdisso. “Sound Radiation from the Boundary in a Circular Lined Duct with Flow”. In: *9th AIAA/CEAS Aeroacoustics Conference and Exhibit* (Apr. 2012). DOI: 10.2514/6.2003-3144. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2003-3144. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2003-3144.

[106] J. Alonso, L. Molisani, and R. Burdisso. “Spectral and Wavenumber Approaches to Obtain Green’s Functions for Convected Wave Equation”. In: *10th AIAA/CEAS Aeroacoustics Conference* (Nov. 2012). DOI: 10.2514/6.2004-2943. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2004-2943. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2004-2943.

[107] S. W. Rienstra and B. J. Tester. “An analytic Green’s function for a lined circular duct containing uniform mean flow”. In: *Journal of Sound and Vibration* 317.3 (2008), pp. 994 –1016. ISSN: 0022-460X. DOI: https://doi.org/10.1016/j.jsv.2008.03.048. URL: http://www.sciencedirect.com/science/article/pii/S0022460X08002836.

[108] Ansys Inc. *Ansys*. 2022. URL: https://www.ansys.com/.

[109] Y. Renard and K. Poulios. “GetFEM: Automated FE Modeling of Multiphysics Problems Based on a Generic Weak Form Language”. In: *ACM Transactions on Mathematical Software* 47 (Dec. 2020), pp. 1–31. DOI: 10.1145/3412849.

[110] COMSOL Inc. *Comsol*. 2022. URL: https://www.comsol.com/.

[111] Y. Yu et al. “Analytical and empirical models for the acoustic dispersion relations in partially filled water pipes”. In: *Applied Acoustics* 179 (2021), p. 108076. ISSN: 0003-682X. DOI: https://doi.org/10.1016/j.apacoust.2021.108076. URL: https://www.sciencedirect.com/science/article/pii/S0003682X21001699.

[112] A. G. Prinn. “On Computing Impulse Responses from Frequency-Domain Finite Element Solutions”. In: *Journal of Theoretical and Computational Acoustics* 29.01 (2021), p. 2050024. DOI: 10.1142/S2591728520500243. URL: https://doi.org/10.1142/S2591728520500243.

[113] I. Farmaga et al. “Evaluation of computational complexity of finite element analysis”. In: *2011 11th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)*. 2011, pp. 213–214.

[114] M. Okereke and S. Keates. *Finite Element Applications. A Practical Guide to the FEM Process.* Springer Cham, 2018, p. 472. ISBN: 978-3-319-67125-3. DOI: 10.1007/978-3-319-67125-3.

[115] J. Melenk and S Sauter. “Convergence analysis for finite element discretizations of the Helmholtz equation with Dirichlet-to-Neumann bound-

ary conditions". In: *Math. Comput.* 79 (Apr. 2010). DOI: 10.1090/S0025-5718-10-02362-8.

[116] L. Maio and P. Fromme. "On ultrasound propagation in composite laminates: advances in numerical simulation". In: *Progress in Aerospace Sciences* 129 (2022). Impact induced dynamic response and failure behavior of aircraft structures, p. 100791. ISSN: 0376-0421. DOI: https://doi.org/10.1016/j.paerosci.2021.100791. URL: https://www.sciencedirect.com/science/article/pii/S0376042121000932.

[117] T. Stepinski, T. Uhl, and W. Staszewski. *Advanced Structural Damage Detection: From Theory to Engineering Applications*. May 2013. ISBN: 9781118422984. DOI: 10.1002/9781118536148.ch4.

[118] J. W. Thomas. *Numerical Partial Differential Equations: Finite Difference Methods*. Texts in Applied Mathematics. Springer New York, 1998. ISBN: 0-387-97999-9.

[119] R. Marklein. "The Finite Integration Technique as a General Tool to Compute Acoustic, Electromagnetic, Elastodynamic, and Coupled Wave Fields". In: Oct. 2002, pp. 201–244. ISBN: 9780471268666.

[120] S. H. Lo. "Finite element mesh generation and adaptive meshing". In: *Progress in Structural Engineering and Materials* 4.4 (2002), pp. 381–399. DOI: https://doi.org/10.1002/pse.135. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/pse.135.

[121] C. Geuzaine and J. F. Remacle. "Gmsh: A 3-D Finite Element Mesh Generator with Built-in Pre- and Post-Processing Facilities". In: *International Journal for Numerical Methods in Engineering* 79 (Sept. 2009), pp. 1309–1331. DOI: 10.1002/nme.2579.

[122] C. Howard and B. Cazzolato. *Acoustic Analyses Using Matlab and Ansys*. Oct. 2014. ISBN: 978-1482223255. DOI: 10.1201/b17825.

[123] J. L. Rose. "The Semi-Analytical Finite Element Method". In: *Ultrasonic Guided Waves in Solid Media*. Cambridge University Press, 2014, 135–154. DOI: 10.1017/CBO9781107273610.011.

[124] R. B. Nelson, S. B. Dong, and R. D. Kalra. “Vibrations and waves in laminated orthotropic circular cylinders”. In: *Journal of Sound Vibration* 18.3 (Oct. 1971), pp. 429–444. DOI: 10.1016/0022-460X(71)90714-0. URL: https://ui.adsabs.harvard.edu/abs/1971JSV....18..429N.

[125] T. Hayashi, K. Koichiro, and J. Rose. “Calculation for Guided Waves in Pipes and Rails”. In: *Key Engineering Materials - KEY ENG MAT* 270 (Aug. 2004), pp. 410–415. DOI: 10.4028/www.scientific.net/KEM.270-273.410.

[126] M. K. Kalkowski, E. Rustighi, and T. P. Waters. “Modelling piezoelectric excitation in waveguides using the semi-analytical finite element method”. In: *Computers Structures* 173 (2016), pp. 174–186. ISSN: 0045-7949. DOI: https://doi.org/10.1016/j.compstruc.2016.05.022. URL: https://www.sciencedirect.com/science/article/pii/S0045794916303558.

[127] I. Bartoli et al. “Modeling wave propagation in damped waveguides of arbitrary cross-section”. In: *Journal of Sound and Vibration* 295.3 (2006), pp. 685–707. ISSN: 0022-460X. DOI: https://doi.org/10.1016/j.jsv.2006.01.021. URL: https://www.sciencedirect.com/science/article/pii/S0022460X06001179.

[128] J. Mu and J. L. Rose. “Guided wave propagation and mode differentiation in hollow cylinders with viscoelastic coatings”. In: *The Journal of the Acoustical Society of America* 124.2 (2008), pp. 866–874. DOI: 10.1121/1.2940586. eprint: https://doi.org/10.1121/1.2940586. URL: https://doi.org/10.1121/1.2940586.

[129] J. J. Ditri and J. L. Rose. “Excitation of guided elastic wave modes in hollow cylinders by applied surface tractions”. In: *Journal of Applied Physics* 72.7 (1992), pp. 2589–2597. DOI: 10.1063/1.351558. eprint: https://doi.org/10.1063/1.351558. URL: https://doi.org/10.1063/1.351558.

[130] K. Umashankar and A. Taflove. “A Novel Method to Analyze Electromagnetic Scattering of Complex Objects”. In: *IEEE Transactions on Electromagnetic Compatibility* EMC-24.4 (1982), pp. 397–405. DOI: 10.1109/TEMC.1982.304054.

[131] Z. A. B. Ahmad and U. Gabbert. “Simulation of Lamb wave reflections at plate edges using the semi-analytical finite element method”. In: *Ultrasonics* 52.7 (2012), pp. 815–820. ISSN: 0041-624X. DOI: https://doi.org/10.1016/j.ultras.2012.05.008. URL: https://www.sciencedirect.com/science/article/pii/S0041624X12001059.

[132] Z. A. B. Ahmad, J. M. Vivar-Perez, and U. Gabbert. “Semi-analytical finite element method for modeling of lamb wave propagation”. In: *CEAS Aeronautical Journal* 4.1 (2013), pp. 21–33. ISSN: 1869-5590. DOI: 10.1007/s13272-012-0056-6. URL: https://doi.org/10.1007/s13272-012-0056-6.

[133] P. Kelly. *Solid Mechanics Part I: An Introduction to Solid Mechanics*. 2021. URL: https://pkel015.connect.amazon.auckland.ac.nz/SolidMechanicsBooks/Part_I/index.html.

[134] P. Kelly. *Solid Mechanics Part II: Engineering Solid Mechanics*. 2021. URL: https://pkel015.connect.amazon.auckland.ac.nz/SolidMechanicsBooks/Part_II/index.html.

[135] F. Jacobsen and P. M. Juhl. *Fundamentals of General Linear Acoustics*. Wiley, 2013. ISBN: 9781118636176. URL: https://books.google.co.uk/books?id=Sq6uFqlHg1gC.

[136] A. Rona. “The Acoustic Resonance of Rectangular and Cylindrical Cavities”. In: *Journal of Algorithms & Computational Technology* 1.3 (2007), pp. 329–356. DOI: 10.1260/174830107782424110. eprint: https://doi.org/10.1260/174830107782424110. URL: https://doi.org/10.1260/174830107782424110.

[137] A. C. Hladky-Hennion. “Finite Element Analysis of the Propagation of Acoustic Waves in Waveguides”. In: *Journal of Sound Vibration* 194.2 (July 1996), pp. 119–136. DOI: 10.1006/jsvi.1996.0349.

[138] Q. Wang, O. Ronneberger, and H. Burkhardt. “Fourier Analysis in Polar and Spherical Coordinates”. In: (2008).

[139] R. L. Herman. *Introduction to partial differential equations*. University of North Carolina Wilmington, 2022.

[140] J. Barshinger, J. L. Rose, and M. J. Avioli. “Guided wave resonance tuning for pipe inspection”. In: *American Society of Mechanical Engineers, Pressure Vessels and Piping Division (Publication) PVP* 450 (Dec. 2002), pp. 49–62. ISSN: 0277-027X. DOI: 10.1115/PVP2002-1635.

[141] R. Kirby and W. Duan. “Guided wave propagation in cylindrical ducts with elastic walls enclosing a fluid moving with a uniform velocity”. In: *Wave Motion* 85 (2018), pp. 1–9.

[142] V. A. Del Grosso. “Analysis of Multimode Acoustic Propagation in Liquid Cylinders with Realistic Boundary Conditions - Application to Sound Speed and Absorption Measurements”. In: *Acta Acustica united with Acustica* 24.6 (June 1971), 299–311(13).

[143] D. C. Gazis. “Three-Dimensional Investigation of the Propagation of Waves in Hollow Circular Cylinders. I. Analytical Foundation”. In: *The Journal of the Acoustical Society of America* 31.5 (1959), pp. 568–573. DOI: 10.1121/1.1907753.

[144] R. M. Mohammad, F. Thabtah, and L. McCluskey. “An assessment of features related to phishing websites using an automated technique”. English. In: *2012 International Conference for Internet Technology and Secured Transactions (ICITST 2012)*. United States: IEEE, Mar. 2013, pp. 492–497. ISBN: 9781467353250.

[145] G. Vrbančič, I. Fister, and V. Podgorelec. “Datasets for phishing websites detection”. In: *Data in Brief* 33 (2020), p. 106438. ISSN: 2352-3409. DOI:

https://doi.org/10.1016/j.dib.2020.106438. URL: https://www.sciencedirect.com/science/article/pii/S2352340920313202.

[146] A. Rogers et al. “RuSentiment: An Enriched Sentiment Analysis Dataset for Social Media in Russian”. In: *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 755–763. URL: https://aclanthology.org/C18-1064.

[147] W. Xu et al. “Extracting Lexically Divergent Paraphrases from Twitter”. In: *Transactions of the Association for Computational Linguistics* 2 (Dec. 2014), pp. 435–448. DOI: 10.1162/tacl_a_00194.

[148] D. Kollias and S. Zafeiriou. “Expression, Affect, Action Unit Recognition: Aff-Wild2, Multi-Task Learning and ArcFace”. In: *BMVC*. 2019.

[149] T. Y. Lin et al. “Microsoft COCO: Common Objects in Context”. In: *Computer Vision – ECCV 2014*. Ed. by D. Fleet et al. Cham: Springer International Publishing, 2014, pp. 740–755. ISBN: 978-3-319-10602-1.

[150] S. Rasp et al. “WeatherBench: A Benchmark Data Set for Data-Driven Weather Forecasting”. In: *Journal of Advances in Modeling Earth Systems* 12.11 (2020), e2020MS002203. DOI: https://doi.org/10.1029/2020MS002203. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2020MS002203. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020MS002203.

[151] D. D. Lucas et al. “Designing optimal greenhouse gas observing networks that consider performance and cost”. In: *Geoscientific Instrumentation, Methods and Data Systems* 4.1 (2015), pp. 121–137. DOI: 10.5194/gi-4-121-2015. URL: https://gi.copernicus.org/articles/4/121/2015/.

[152] T. Komatsu et al. “Acoustic Event Detection Method Using Semi-Supervised Non-Negative Matrix Factorization with Mixtures of Local Dictionaries”. In: Sept. 2016.

[153] H. He and E. A. Garcia. “Learning from Imbalanced Data”. In: *IEEE Transactions on Knowledge and Data Engineering* 21.9 (2009), pp. 1263–1284. DOI: 10.1109/TKDE.2008.239.

[154] E. J. Humphrey et al. “JAMS: A JSON annotated music specification for reproducible MIR research”. English (US). In: 2014, pp. 591–596.

[155] B. Gygi and V. Shafiro. “Environmental sound research as it stands today”. In: *The Journal of the Acoustical Society of America* 121 (Feb. 2007), p. 3165. DOI: 10.1121/1.4782254.

[156] A. Dang, T. H. Vu, and J. C. Wang. “A survey of deep learning for polyphonic sound event detection”. In: *2017 International Conference on Orange Technologies (ICOT)*. 2017, pp. 75–78. DOI: 10.1109/ICOT.2017.8336092.

[157] M. A. Hossan, S. Memon, and M. A. Gregory. “A novel approach for MFCC feature extraction”. In: *2010 4th International Conference on Signal Processing and Communication Systems*. 2010, pp. 1–5. DOI: 10.1109/ICSPCS.2010.5709752.

[158] S. Gupta et al. “Feature Extraction Using Mfcc”. In: *Signal Image Processing : An International Journal* 4 (Aug. 2013), pp. 101–108. DOI: 10.5121/sipij.2013.4408.

[159] M. S. Likitha et al. “Speech based human emotion recognition using MFCC”. In: *2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*. 2017, pp. 2257–2260. DOI: 10.1109/WiSPNET.2017.8300161.

[160] T. Majtner, S. Yildirim-Yayilgan, and J. Y. Hardeberg. “Combining deep learning and hand-crafted features for skin lesion classification”. In: *2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA)*. 2016, pp. 1–6. DOI: 10.1109/IPTA.2016.7821017.

[161] D. de Benito-Gorron, D. Ramos, and D. T. Toledano. “A Multi-Resolution Approach to Sound Event Detection in DCASE 2020 Task4”. In: *Proceed-*

ings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020). Tokyo, Japan, Nov. 2020, pp. 36–40.

[162] C. Duxbury et al. “A comparison between fixed and multiresolution analysis for onset detection in musical signals”. In: *Proceedings of the International Conference on Digital Audio Effects, DAFx* (2004), pp. 207–211. ISSN: 2413-6700.

[163] K. M. M. Prabhu. *Window Functions and Their Applications in Signal Processing (1st ed.)* Ed. by CRC Press. DOI: https://doi.org/10.1201/9781315216386.

[164] K. Paliwal, J. Lyons, and K. Wojcicki. “Preference for 20-40 ms window duration in speech analysis”. In: Jan. 2011, pp. 1 –4. DOI: 10.1109/ICSPCS.2010.5709770.

[165] C. Rusu, B. Dumitrescu, and S. A. Tsaftaris. “Explicit Shift-Invariant Dictionary Learning”. In: *IEEE Signal Processing Letters* 21.1 (2014), pp. 6–9. DOI: 10.1109/LSP.2013.2288788.

[166] F. Champagnat and C. Herzet. “Translation-invariant interpolation of parametric dictionaries”. In: *iTwist 2020 - International Traveling Workshop on Interactions between low-complexity data models and Sensing Techniques*. Nantes, France, Dec. 2020, pp. 1–3.

[167] M. Welling. “Robust Higher Order Statistics”. In: *AISTATS*. 2005.

[168] D. Campo, O. L. Quintero, and M. Bastidas. “Multiresolution analysis (discrete wavelet transform) through Daubechies family for emotion recognition in speech.” In: *Journal of Physics: Conference Series* 705 (Apr. 2016), p. 012034. DOI: 10.1088/1742-6596/705/1/012034. URL: https://doi.org/10.1088/1742-6596/705/1/012034.

[169] Y. Ando. “Autocorrelation-based features for speech representation”. In: *Proceedings of Meetings on Acoustics* 19.1 (2013), p. 060033. DOI: 10.1121/1.4795143. eprint: https://asa.scitation.org/doi/pdf/10.1121/1.4795143. URL: https://asa.scitation.org/doi/abs/10.1121/1.4795143.

[170] K. M. Ravikumar. “Acoustic Noise Classification and Characterization Using Statistical Properties”. In: 2014.

[171] H. Fletcher. “Auditory patterns”. In: *Reviews of modern physics* 12.1 (1940), p. 47.

[172] E. Zwicker and E. Terhardt. “Analytical expressions for critical-band rate and critical bandwidth as a function of frequency”. In: *The Journal of the Acoustical Society of America* 68.5 (1980), pp. 1523–1525. DOI: 10.1121/1.385079. eprint: https://doi.org/10.1121/1.385079. URL: https://doi.org/10.1121/1.385079.

[173] A. Aertsen and P. I. M. Johannesma. “Spectro-temporal receptive fields of auditory neurons in the grassfrog - I. Characterization of tonal and natural stimuli”. In: *Biological Cybernetics* 38 (Nov. 1980), pp. 223–234. DOI: 10.1007/BF00337015.

[174] S. S. Stevens, J. Volkmann, and E. B. Newman. “A Scale for the Measurement of the Psychological Magnitude Pitch”. In: *The Journal of the Acoustical Society of America* 8.3 (1937), pp. 185–190. DOI: 10.1121/1.1915893. eprint: https://doi.org/10.1121/1.1915893. URL: https://doi.org/10.1121/1.1915893.

[175] S. K. Kopparapu and M. Laxminarayana. “Choice of Mel filter bank in computing MFCC of a resampled speech”. In: *10th International Conference on Information Science, Signal Processing and their Applications (ISSPA 2010)*. 2010, pp. 121–124. DOI: 10.1109/ISSPA.2010.5605491.

[176] J. Harrington and S. Cassidy. *The Physics of Speech*. Dordrecht: Springer Netherlands, 1999, pp. 9–28. ISBN: 978-94-011-4657-9. DOI: 10.1007/978-94-011-4657-9_2. URL: https://doi.org/10.1007/978-94-011-4657-9_2.

[177] X. Huang et al. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall PTR, 2001. ISBN: 9780130226167.

[178] J. Morlet et al. “Wave propagation and sampling theory—Part I: Complex signal and scattering in multilayered media”. In: *GEOPHYSICS* 47.2 (1982), pp. 203–221. DOI: 10.1190/1.1441328. eprint: https://doi.org/10.1190/1.1441328. URL: https://doi.org/10.1190/1.1441328.

[179] J. C. van den Berg. *Wavelets in Physics*. Cambridge University Press, 1999. DOI: 10.1017/CBO9780511613265.

[180] M. Unser and A. Aldroubi. “A review of wavelets in biomedical applications”. In: *Proceedings of the IEEE* 84.4 (1996), pp. 626–638. DOI: 10.1109/5.488704.

[181] M. Grimaldi, P. Cunningham, and A. Kokaram. “A Wavelet Packet Representation of Audio Signals for Music Genre Classification Using Different Ensemble and Feature Selection Techniques”. In: MIR ’03. Berkeley, California: Association for Computing Machinery, 2003, 102–108. ISBN: 1581137788. DOI: 10.1145/973264.973281. URL: https://doi.org/10.1145/973264.973281.

[182] J. Bruna and S. Mallat. “Invariant Scattering Convolution Networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8 (2013), pp. 1872–1886. DOI: 10.1109/TPAMI.2012.230.

[183] S. Mallat. “Understanding deep convolutional networks”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065 (2016), p. 20150203. DOI: 10.1098/rsta.2015.0203. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.2015.0203. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2015.0203.

[184] A. Mertins. *Signal Analysis: Wavelets, Filter Banks, Time-Frequency Transforms and Applications*. Ultrasound in Biomedicine Research Series. Wiley, 1999. ISBN: 9780471986263. URL: https://books.google.co.uk/books?id=hctqQgAACAAJ.

[185] S. Schimmel and L. Atlas. “Coherent envelope detection for modulation filtering of speech”. In: *Proceedings. (ICASSP ’05). IEEE International*

Conference on Acoustics, Speech, and Signal Processing, 2005. Vol. 1. 2005, I/221–I/224 Vol. 1. DOI: 10.1109/ICASSP.2005.1415090.

[186] S. Mallat. "A theory for multiresolution signal decomposition: the wavelet representation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11.7 (1989), pp. 674–693. DOI: 10.1109/34.192463.

[187] V. Lostanlen. *Scattering.m: a MATLAB toolbox for signal scattering*. 2021. URL: https://github.com/lostanlen/scattering.m.

[188] I. Waldspurger. "Phase Retrieval for Wavelet Transforms". In: *IEEE Transactions on Information Theory* 63.5 (2017), pp. 2993–3009. DOI: 10.1109/TIT.2017.2672727.

[189] Y. Bengio and Y. Lecun. "Convolutional Networks for Images, Speech, and Time-Series". In: (Nov. 1997).

[190] *jpeg2000*. 2021. URL: https://jpeg.org/jpeg2000/.

[191] H. G. Musmann. "Genesis of the MP3 audio coding standard". In: *IEEE Transactions on Consumer Electronics* 52.3 (2006), pp. 1043–1049. DOI: 10.1109/TCE.2006.1706505.

[192] C. Z. Di et al. "Multilevel functional principal component analysis". In: *The annals of applied statistics* 3.1 (2009), 458–488. DOI: https://doi.org/10.1214/08-AOAS206SUPP.

[193] F. De la Torre and M. J. Black. "Robust principal component analysis for computer vision". In: *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. Vol. 1. 2001, 362–369 vol.1. DOI: 10.1109/ICCV.2001.937541.

[194] J. Le-Rademacher and L. Billard. "Symbolic Covariance Principal Component Analysis and Visualization for Interval-Valued Data". In: *Journal of Computational and Graphical Statistics* 21.2 (2012), pp. 413–432. DOI: 10.1080/10618600.2012.679895. eprint: https://doi.org/10.1080/10618600.2012.679895. URL: https://doi.org/10.1080/10618600.2012.679895.

[195] I. T. Jolliffe and J. Cadima. “Principal component analysis: a review and recent developments”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065 (2016), p. 20150202. DOI: 10.1098/rsta.2015.0202. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.2015.0202. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2015.0202.

[196] K. I. Park. *Random Variables*. Cham: Springer International Publishing, 2018. ISBN: 978-3-319-68075-0. DOI: 10.1007/978-3-319-68075-0_4. URL: https://doi.org/10.1007/978-3-319-68075-0_4.

[197] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning series. MIT Press, 2012. ISBN: 9780262018029. URL: https://books.google.co.uk/books?id=NZP6AQAAQBAJ.

[198] B. Schölkopf, A. Smola, and K. R. Müller. “Nonlinear Component Analysis as a Kernel Eigenvalue Problem”. In: *Neural Computation* 10.5 (July 1998), pp. 1299–1319. ISSN: 0899-7667. DOI: 10.1162/089976698300017467. eprint: https://direct.mit.edu/neco/article-pdf/10/5/1299/813905/089976698300017467.pdf. URL: https://doi.org/10.1162/089976698300017467.

[199] B. Ghojogh et al. “Reproducing Kernel Hilbert Space, Mercer’s Theorem, Eigenfunctions, Nyström Method, and Use of Kernels in Machine Learning: Tutorial and Survey”. In: *CoRR* abs/2106.08443 (2021). arXiv: 2106.08443. URL: https://arxiv.org/abs/2106.08443.

[200] M. Gönen and E. Alpaydın. “Multiple Kernel Learning Algorithms.” In: *Journal of Machine Learning Research* 12 (July 2011), pp. 2211–2268.

[201] L Breiman. “Random Forests”. In: *Machine Learning* 45 (Oct. 2001), pp. 5–32. DOI: 10.1023/A:1010950718922.

[202] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. DOI: 10.1017/CBO9780511809071.

[203] B. Mor, S. Garhwal, and A. Kumar. "A Systematic Review of Hidden Markov Models and Their Applications". In: *Archives of Computational Methods in Engineering* 28 (2020), pp. 1429–1448.

[204] T. Cover and P. Hart. "Nearest neighbor pattern classification". In: *IEEE Transactions on Information Theory* 13.1 (1967), pp. 21–27. DOI: 10.1109/TIT.1967.1053964.

[205] I. Paryudi. "What Affects K Value Selection In K-Nearest Neighbor". In: *International Journal of Scientific & Technology Research* 8 (2019), pp. 86–92.

[206] D. Coomans and D. L. Massart. "Alternative k-nearest neighbour rules in supervised pattern recognition: Part 1. k-Nearest neighbour classification by using alternative voting rules". In: *Analytica Chimica Acta* 136 (1982), pp. 15–27. ISSN: 0003-2670. DOI: https://doi.org/10.1016/S0003-2670(01)95359-0. URL: https://www.sciencedirect.com/science/article/pii/S0003267001953590.

[207] M. Beckmann, N. Ebecken, and B. Lima. "A KNN Undersampling Approach for Data Balancing". In: *Journal of Intelligent Learning Systems and Applications* 7 (Nov. 2015), pp. 104–116. DOI: 10.4236/jilsa.2015.74010.

[208] J. Grus. *Data Science from Scratch: First Principles with Python*. O'Reilly Media, 2019. ISBN: 9781492041108.

[209] S. An et al. "Data reduction based on NN-kNN measure for NN classification and regression". In: *International Journal of Machine Learning and Cybernetics* (Apr. 2021), pp. 1–17. DOI: 10.1007/s13042-021-01327-3.

[210] S. Garcia et al. "Prototype Selection for Nearest Neighbor Classification: Taxonomy and Empirical Study". In: *IEEE Transactions on Pat-*

tern Analysis and Machine Intelligence 34.3 (2012), pp. 417–435. DOI: 10.1109/TPAMI.2011.142.

[211] C. A. Bhardwaj, M. Mishra, and K. Desikan. “Dynamic Feature Scaling for K-Nearest Neighbor Algorithm”. In: *ArXiv* abs/1811.05062 (2017).

[212] K. Q. Weinberger, J. Blitzer, and L. Saul. “Distance Metric Learning for Large Margin Nearest Neighbor Classification”. In: *Advances in Neural Information Processing Systems*. Ed. by Y. Weiss, B. Schölkopf, and J. Platt. Vol. 18. MIT Press, 2006.

[213] J. Yu et al. “Distance Learning for Similarity Estimation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.3 (2008), pp. 451–462. DOI: 10.1109/TPAMI.2007.70714.

[214] M. Luptáčik and M. Luptácik. “Kuhn–Tucker Conditions”. In: vol. 36. Sept. 2010, pp. 25–58. ISBN: 978-0-387-89551-2. DOI: 10.1007/978-0-387-89552-9_2.

[215] J. Platt. “Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines”. In: *Advances in Kernel Methods-Support Vector Learning* 208 (July 1998).

[216] G. Wang. “A Survey on Training Algorithms for Support Vector Machine Classifiers”. In: *2008 Fourth International Conference on Networked Computing and Advanced Information Management*. Vol. 1. 2008, pp. 123–128. DOI: 10.1109/NCM.2008.103.

[217] S. Shalev-Shwartz et al. “Pegasos: primal estimated sub-gradient solver for SVM”. In: *Mathematical Programming* 127 (2011), pp. 3–30.

[218] C. J. Hsieh et al. “A Dual Coordinate Descent Method for Large-Scale Linear SVM”. In: *Proceedings of the 25th International Conference on Machine Learning*. ICML ’08. Helsinki, Finland: Association for Computing Machinery, 2008, 408–415. ISBN: 9781605582054. DOI: 10.1145/1390156.1390208. URL: https://doi.org/10.1145/1390156.1390208.

[219] Y. J. Lee and S. Y. Huang. “Reduced Support Vector Machines: A Statistical Theory”. In: *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council* 18 (Feb. 2007), pp. 1–13. DOI: 10.1109/TNN.2006.883722.

[220] J. Cervantes, X. Li, and W. Yu. “Support Vector Machine Classification Based on Fuzzy Clustering for Large Data Sets”. In: vol. 4293. Nov. 2006, pp. 572–582. ISBN: 978-3-540-49026-5. DOI: 10.1007/11925231_54.

[221] B. E. Boser, I. Guyon, and V. N. Vapnik. “A training algorithm for optimal margin classifiers”. In: *COLT ’92*. 1992.

[222] J. Cervantes et al. “A comprehensive survey on support vector machine classification: Applications, challenges and trends”. In: *Neurocomputing* 408 (May 2020). DOI: 10.1016/j.neucom.2019.10.118.

[223] R. Akbani, S. Kwek, and N. Japkowicz. “Applying Support Vector Machines to Imbalanced Data Sets”. In: vol. 3201. Sept. 2004, pp. 39–50. ISBN: 978-3-540-23105-9. DOI: 10.1007/978-3-540-30115-8_7.

[224] Y. Liu et al. “Combining integrated sampling with SVM ensembles for learning from imbalanced datasets”. In: *Inf. Process. Manage.* 47 (July 2011), pp. 617–631. DOI: 10.1016/j.ipm.2010.11.007.

[225] L. A. Gottlieb, E. Kaufman, and A. Kontorovich. “Apportioned Margin Approach for Cost Sensitive Large Margin Classifiers”. In: *ArXiv* abs/2002.01408 (2021).

[226] S. Escalera, O. Pujol, and P. Radeva. “On the Decoding Process in Ternary Error-Correcting Output Codes”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.1 (2010), pp. 120–134. DOI: 10.1109/TPAMI.2008.266.

[227] O. Maimon and L. Rokach. *Introduction to Soft Computing for Knowledge Discovery and Data Mining*. Jan. 2008. ISBN: 978-0-387-69934-9. DOI: 10.1007/978-0-387-69935-6_1.

[228] A. Mesaros, T. Heittola, and T. Virtanen. “Metrics for Polyphonic Sound Event Detection”. In: *Applied Sciences* 6.6 (2016). ISSN: 2076-3417. DOI: 10.3390/app6060162. URL: https://www.mdpi.com/2076-3417/6/6/162.

[229] D. C. Blair. “Information Retrieval, 2nd ed. C.J. Van Rijsbergen. London: Butterworths; 1979: 208 pp. Price: $32.50”. In: *Journal of the American Society for Information Science* 30.6 (1979), pp. 374–375. DOI: https://doi.org/10.1002/asi.4630300621. eprint: https://asistdl.onlinelibrary.wiley.com/doi/pdf/10.1002/asi.4630300621. URL: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/asi.4630300621.

[230] W. Li et al. “Theoretical and numerical investigations of the nonlinear acoustic response of feature guided waves in a welded joint”. In: *Wave Motion* 101 (Dec. 2020).

[231] A. M. Zelenyak, M. A. Hamstad, and M. G. R. Sause. “Modeling of Acoustic Emission Signal Propagation in Waveguides”. In: *Sensors* 15.5 (2015), pp. 11805–11822. ISSN: 1424-8220. DOI: 10.3390/s150511805. URL: https://www.mdpi.com/1424-8220/15/5/11805.

[232] I. Bartoli et al. “Modeling wave propagation in damped waveguides of arbitrary cross-section”. In: *Journal of Sound and Vibration* 295.3 (2006), pp. 685–707. ISSN: 0022-460X. DOI: https://doi.org/10.1016/j.jsv.2006.01.021. URL: https://www.sciencedirect.com/science/article/pii/S0022460X06001179.

[233] P. Zuo, Y. Zhou, and Z. Fan. “Numerical studies of nonlinear ultrasonic guided waves in uniform waveguides with arbitrary cross sections”. In: *AIP Advances* 6.7 (2016), p. 075207. DOI: 10.1063/1.4959005.

[234] D. C. Gazis. “Three-Dimensional Investigation of the Propagation of Waves in Hollow Circular Cylinders. II. Numerical Results”. In: *The Journal of the Acoustical Society of America* 31.5 (1959), pp. 573–578. DOI: 10.1121/1.1907754.

[235] J. A. F. Santiago and L. C. Wrobel. “An efficient Green’s function for acoustic waveguide problems”. In: *Communications in Numerical Methods in Engineering* 23.7 (2007), pp. 703–719. DOI: https://doi.org/10.1002/cnm.921. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cnm.921. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cnm.921.

[236] A. Snakowska and R. Wyrzykowski. “Calculation of the acoustical field of a semi-infinite cylindrical wave-guide by means of the Green function expressed in cylindrical coordinates.” In: 11 (Jan. 1986), pp. 261–285.

[237] J. Backman. “A model of open-baffle loudspeakers”. In: *Journal of the audio engineering society* (Sept. 1999).

[238] *Visaton: k-50-wp-8-ohm speaker*. URL: https://www.visaton.de/en/products/drivers/fullrange-systems/k-50-wp-8-ohm.

[239] F. Anselmet and P. O. Mattei. *Acoustics, Aeroacoustics and Vibrations*. John Wiley and Sons, 2016. ISBN: 978-1-84821-861-1.

[240] R. Priemer. *MATLAB® for Electrical and Computer Engineering Students and Professionals - With Simulink®*. Institution of Engineering and Technology, 2013. ISBN: 978-1-61353-188-4.

[241] P. C. Hansen. “Truncated Singular Value Decomposition Solutions to Discrete Ill-Posed Problems with Ill-Determined Numerical Rank”. In: *SIAM Journal on Scientific and Statistical Computing* 11.3 (Dec. 1990), pp. 503–518. DOI: 10.1137/0911028. eprint: https://doi.org/10.1137/0911028. URL: https://doi.org/10.1137/0911028.

[242] A. Neumaier. “Solving Ill-Conditioned and Singular Linear Systems: A Tutorial on Regularization”. In: *SIAM Review* 40.3 (1998), pp. 636–666. DOI: 10.1137/S0036144597321909. eprint: https://doi.org/10.1137/S0036144597321909. URL: https://doi.org/10.1137/S0036144597321909.

[243] S. Noschese and L. Reichel. “A modified truncated singular value decomposition method for discrete ill-posed problems”. In: *Numerical Linear Algebra with Applications* 21 (Dec. 2014). DOI: 10.1002/nla.1938.

[244] J. C. A. Barata and M. S. Hussein. “The Moore–Penrose Pseudoinverse: A Tutorial Review of the Theory”. In: *Brazilian Journal of Physics* 42.1-2 (2011), pp. 146–165. DOI: 10.1007/s13538-011-0052-z.

[245] J. W. Demmel. “On Condition Numbers and the Distance to the Nearest Ill-posed Problem.” In: *Numerische Mathematik* 51 (1987), pp. 251–290. URL: http://eudml.org/doc/133195.

[246] W. Fladung and R. Rost. “Application and correction of the exponential window for frequency response functions”. In: *Mechanical Systems and Signal Processing* 11.1 (1997), pp. 23–36. DOI: https://doi.org/10.1006/mssp.1996.0084. URL: https://www.sciencedirect.com/science/article/pii/S0888327096900849.

[247] F. Font, G. Roma, and X. Serra. “Freesound Technical Demo”. In: *Proceedings of the 21st ACM International Conference on Multimedia*. MM ’13. Barcelona, Spain: Association for Computing Machinery, 2013, 411–412. ISBN: 9781450324045. DOI: 10.1145/2502081.2502245. URL: https://doi.org/10.1145/2502081.2502245.

[248] *AUDIOSET*. URL: https://research.google.com/audioset/index.html.

[249] N. Turpault and R. Serizel. *DESED_synthetic*. Version v3.0. Mar. 2020. DOI: 10.5281/zenodo.4569096. URL: https://doi.org/10.5281/zenodo.4569096.

[250] *Reverb challenge*. URL: https://reverb2014.dereverberation.com/index.html.

[251] *DCASE*. URL: http://dcase.community/.

[252] G. Lafay et al. “A Morphological Model for Simulating Acoustic Scenes and Its Application to Sound Event Detection”. In: *IEEE/ACM Trans. Audio, Speech and Lang. Proc.* 24.10 (Oct. 2016), 1854–1864. ISSN: 2329-9290. DOI: 10.1109/TASLP.2016.2587218. URL: https://doi.org/10.1109/TASLP.2016.2587218.

[253] *Brüel & Kjær: 8103 hydrophone*. URL: https://www.bksv.com/en/transducers/acoustic/microphones/hydrophones/8103.

[254] *Brüel & Kjær: Nexus 2692 amplifier*. URL: https://www.bksv.com/en/transducers/signal-conditioning/charge/2692-a-011.

[255] *Adlink: USB-1210 ADC*. URL: https://www.adlinktech.com/Products/Data_Acquisition/USBDAQ/USB-1210?lang=en.

[256] S. Liebich et al. “Active noise cancellation in headphones by digital robust feedback control”. In: *2016 24th European Signal Processing Conference (EUSIPCO)*. 2016, pp. 1843–1847. DOI: 10.1109/EUSIPCO.2016.7760567.

[257] M. Gabrea, E. Mandridake, and M. Najim. “A single microphone noise canceller based on adaptive Kalman filter”. In: *1996 8th European Signal Processing Conference (EUSIPCO 1996)*. 1996, pp. 1–4.

[258] R. C. Maher. “Overview of Audio Forensics”. In: *Intelligent Multimedia Analysis for Security Applications*. Ed. by Husrev Taha Sencar et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. ISBN: 978-3-642-11756-5. DOI: 10.1007/978-3-642-11756-5_6. URL: https://doi.org/10.1007/978-3-642-11756-5_6.

[259] J. Xie, J. Colonna, and J. Zhang. “Bioacoustic signal denoising: a review”. In: *Artificial Intelligence Review* 54 (June 2021), pp. 1–23. DOI: 10.1007/s10462-020-09932-4.

[260] B. Wang and W. Wong. “Real time hearing enhancement in crowded social environments with noise gating”. In: *Speech Communication* 99 (2018), pp. 173–182. ISSN: 0167-6393. DOI: https://doi.org/10.

1016/j.specom.2018.03.010. URL: https://www.sciencedirect.com/science/article/pii/S0167639317303527.

[261] Audacity Team. *Audacity(R): Free Audio Editor and Recorder*. 2021. URL: www.audacityteam.org/.

[262] P. Sadovsky and B. Karel. "Optimisation of the Transient response of a Digital Filter". In: *Radioengineering* 9 (2000).

[263] M. Frigo and S. G. Johnson. "The Design and Implementation of FFTW3". In: *Proceedings of the IEEE* 93.2 (2005). Special issue on "Program Generation, Optimization, and Platform Adaptation", pp. 216–231.

[264] *Urban-SED*. URL: http://urbansed.weebly.com/.

[265] J. Salamon and J. P. Bello. "Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification". In: *IEEE Signal Processing Letters* 24.3 (2017), pp. 279–283. DOI: 10.1109/LSP.2017.2657381.

[266] B. McFee, E. J. Humphrey, and J. Bello. "A Software Framework for Musical Data Augmentation". In: *ISMIR*. 2015.

[267] Dolby E Bitstreams. "Standards and Practices for Authoring Dolby Digital and Dolby E Bitstreams". In: 2002.

[268] G. Parascandolo, H. Huttunen, and T. Virtanen. "Recurrent neural networks for polyphonic sound event detection in real life recordings". In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2016, pp. 6440–6444. DOI: 10.1109/ICASSP.2016.7472917.

[269] N. Takahashi et al. "Deep Convolutional Neural Networks and Data Augmentation for Acoustic Event Detection". In: *CoRR* abs/1604.07160 (2016). arXiv: 1604.07160. URL: http://arxiv.org/abs/1604.07160.

[270] J. Driedger and M. Müller. "A Review of TimeScale Modification of Music Signals". In: *Applied Sciences* 6.2 (2016). ISSN: 2076-3417. DOI: 10.3390/app6020057. URL: https://www.mdpi.com/2076-3417/6/2/57.

[271] C. Mckay et al. “ACE: A framework for optimizing music classification”. In: *Proceedings of the International Conference on Music Information Retrieval*. 2005.

[272] C. Cannam et al. “The Sonic Visualiser: A Visualisation Platform for Semantic Descriptors from Musical Signals”. In: *ISMIR*. 2006.

[273] B. McFee et al. “Pump up the JAMS : V0.2 and beyond”. In: 2015.

[274] J. Salamon, C. Jacoby, and J. P. Bello. “A Dataset and Taxonomy for Urban Sound Research”. In: *22nd ACM International Conference on Multimedia (ACM-MM’14)*. Orlando, FL, USA, Nov. 2014, pp. 1041–1044.

[275] *JAMS*. URL: https://github.com/marl/jams.

[276] P. J. Lee et al. “Leak location in pipelines using the impulse response function”. In: *Journal of Hydraulic Research* 45.5 (2007), pp. 643–652. DOI: 10.1080/00221686.2007.9521800.

[277] D. Król and R. Wielgat. “Enhancement of loudspeaker impulse response measurement using beamforming methods”. In: *2012 International Conference on Signals and Electronic Systems (ICSES)*. 2012, pp. 1–5. DOI: 10.1109/ICSES.2012.6382255.

[278] D. Diaz-Guerra, A. Miguel, and J. R. Beltran. “gpuRIR: A python library for room impulse response simulation with GPU acceleration”. In: *Multimedia Tools and Applications* 80.4 (Oct. 2020), 5653–5671. ISSN: 1573-7721. DOI: 10.1007/s11042-020-09905-3. URL: http://dx.doi.org/10.1007/s11042-020-09905-3.

[279] H. Zhivomirov. “On the development of STFT-analysis and ISTFT-synthesis routines and their practical implementation”. In: *TEM Journal* 8 (Feb. 2019), pp. 56–64. DOI: 10.18421/TEM81-07.

[280] Mathwork Inc. *Matlab: stft*. 2021. URL: https://www.mathworks.com/help/signal/ref/stft.html.

[281] Z. Nematzadeh, R. Ibrahim, and A. Selamat. “Comparative studies on breast cancer classifications with k-fold cross validations using machine learning techniques”. In: *2015 10th Asian Control Conference (ASCC)*. 2015, pp. 1–6. DOI: 10.1109/ASCC.2015.7244654.

[282] M. Lorbach et al. “Learning to recognize rat social behavior: Novel dataset and cross-dataset application”. In: *Journal of Neuroscience Methods* 300 (2018). Measuring Behaviour 2016, pp. 166–172. ISSN: 0165-0270. DOI: https://doi.org/10.1016/j.jneumeth.2017.05.006. URL: https://www.sciencedirect.com/science/article/pii/S0165027017301255.

[283] A. Vabalas et al. “Machine learning algorithm validation with a limited sample size”. In: *PLOS ONE* 14 (Nov. 2019), e0224365. DOI: 10.1371/journal.pone.0224365.

[284] G. Forman and M. Scholz. “Apples-to-Apples in Cross-Validation Studies: Pitfalls in Classifier Performance Measurement ABSTRACT”. In: *SIGKDD Explorations* 12 (Jan. 2010), pp. 49–57.

[285] S. Varma and R. Simon. “Bias in Error Estimation When Using Cross-Validation for Model Selection.” BMC Bioinformatics, 7(1), 91”. In: *BMC bioinformatics* 7 (Feb. 2006), p. 91. DOI: 10.1186/1471-2105-7-91.

[286] D. Michelsanti et al. *An Overview of Deep-Learning-Based Audio-Visual Speech Enhancement and Separation*. 2020. DOI: 10.48550/ARXIV.2008.09586. URL: https://arxiv.org/abs/2008.09586.

[287] B. McFee et al. “librosa: Audio and Music Signal Analysis in Python”. In: Jan. 2015, pp. 18–24. DOI: 10.25080/Majora-7b98e3ed-003.

[288] I. Kavalerov et al. “Universal Sound Separation”. In: *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. 2019, pp. 175–179. DOI: 10.1109/WASPAA.2019.8937253.

[289] Mathwork Inc. *Matlab: programmatic fitting*. 2021. URL: https://uk.mathworks.com/help/matlab/data_analysis/programmatic-fitting.html.

[290] J. Andén et al. *Scatnet*. 2021. URL: http://www.di.ens.fr/data/software/scatnet/.

[291] L. Sifre et al. *ScatNet : a MATLAB Toolbox for Scattering Networks*. English. 2013.