

# Sound Analysis, Synthesis and Processing

Augusto Sarti  
Oliviero Massi

Paolo Bestagini  
Mirco Pezzoli

Fabio Antonacci  
Federico Miotello

Academic Year 2023-2024  
Music and Acoustic Engineering  
Politecnico di Milano

This notebook has been made with the contribution of:

On the cover:

Jean-Baptiste Camille Corot, *Orphée ramenant Eurydice des enfers*, 1861, oil on canvas,  
112,3x137,1 cm, Museum of Fine Arts, Houston

## Index - Digital Audio Analysis and Processing

Introduction to the course .....	1
Sound Analysis.....	2
Short Time Fourier Transform.....	2
Resolving sinusoidal peaks using STFT.....	4
Resolution.....	8
Spectral interpolation.....	9
From analysis to synthesis.....	14
Signal modelling on Matlab .....	25
Basical operations with audio: .....	26
Analysis of the signal: .....	26
Modelling the noise .....	28
Convolution .....	29
Statistics .....	30
Exercise 3 - Simulate additive noise.....	34
Exercise 4 - Frequency analysis.....	36
Overlap and add .....	42
Filtering of indefinite length signals.....	48
Wiener Filtering .....	63
Graphical interpretation .....	69
Audio Restoration .....	72
Tape recording - A little bit of history.....	73
Formats for studio recording .....	81
How a tape works .....	81
Distortion .....	83
Tape Speed .....	83
Dolby .....	84
DBX compressor/expander .....	85
Impulse noise in tapes .....	85
Phonographs recording - A little bit of history.....	85
Phonographs recording – history of formats.....	86
Stereo recording – How it works on vinyl records.....	90
Degradation causes .....	90
Audio CD .....	91
Data reading.....	92
Digital Versatile Disc – DVD.....	92
Blu-ray disc .....	94
Comparison of different disc characteristics .....	94
Recorder Characteristics .....	95
Compatibility .....	95
Protection system.....	95
Enhancements .....	95
Degradation in DVD and CD .....	95
Digital Audio Restoration .....	96
Recap and in-depth analysis on Wiener Filter .....	97
Implementation and performance .....	98
Characterisation .....	98
Alternative formulation.....	99
Power spectral matching .....	99
Parametric Wiener Filtering.....	100

Background noise reduction .....	100
Short-Time Spectral Attenuation .....	100
Wiener Filter and PSM filter implemented on MATLAB for noise reduction .....	101
STSA on MATLAB for noise reduction .....	104
Dereverberation .....	106
Local degradations .....	109
Local degradation in the context of historical restauration .....	110
A General Scheme for Detection and Estimation .....	115
Linear Predicting Coding .....	126
Main idea: .....	126
How to achieve it: .....	126
Statistical interpretation of the infinite memory linear prediction .....	129
Recreate the signal from filter coefficients and white noise .....	130
Dealing with non-stationary signal .....	132
Application of LPC: .....	136
Time and Pitch Scaling .....	138
Time Scaling .....	139
Pitch Scaling .....	139
Pitch and Frequency Scaling .....	140
Duality principle .....	140
Methods to perform time and pitch scaling: .....	141
Parametric Frequency-based methods .....	141
Non-Parametric Frequency-based methods .....	142
Analogue Pitch Scaling .....	143
Analogical Time Scaling .....	144
Digital Counterpart .....	144
Time-domain non-parametric techniques (waveform similarity) .....	146
SOLA: .....	147
PSOLA: .....	150
Frequency-domain non-parametric techniques (STFT-based) .....	151
Matlab's lesson upon pitch and time rescaling .....	157
Application for Adaptive Filtering .....	160
Noise Cancellation .....	160
Acoustic echo Cancellation .....	161
Background noise removal .....	163
Active noise control .....	166
Features and Classification .....	167
Feature extraction .....	168
Geometric interpretation .....	171
Audio Features .....	172
Basic descriptors .....	172
Harmonicity Descriptor .....	176
Timbral Descriptors .....	176
Other descriptors .....	178
Classification methods .....	179
Feature extraction and classification on MATLAB .....	181
Exercise 1 – Feature Extraction .....	181
Exercise 2 – KNN Classification .....	183

Microphone arrays - Spatial methods for DOA estimation.....	189
Setting.....	189
Data Model .....	190
How to derive $\theta$ .....	196
Non-parametric method (Spatial Filtering) .....	196
Delay-and-sum beamformer .....	198
Pseudo-spectrum .....	199
Resolution.....	200
Capon Method .....	201
Parametric methods .....	201
Covariance model .....	202
MUSIC (MULTiple SIgnal Classification) .....	210
ESPRIT (Estimation of Signal Parameters via Rotational Invariance Technique) .....	211
Localisation of wide-band sources .....	213
Data Model .....	213
Time Difference of Arrival (TDOA) .....	213
Generalised cross-correlation (GCC) method.....	215
Adaptive Eigenvalue Decomposition (AED) method .....	217
Source Separation.....	220
Source separation using binary masks.....	221
Methodology .....	222
Independent Component Analysis (ICA).....	225
Kurtosis .....	230
Negentropy .....	231
Mutual Information .....	231
Kullback-Leibler divergence .....	232
Exercises .....	233
Exercise on Wiener Filtering .....	233
Exercises related to windowing .....	238

## Index - Sound Synthesis and Spatial Processing

Quick and (very) dirty recap on acoustics.....	241
Power of instruments in an orchestra.....	242
Intensity.....	242
Envelope.....	243
Timbre.....	243
Acoustic waves .....	244
Diffusion and Attenuation.....	244
Reflection.....	244
Absorption.....	245
Atmospheric effects.....	245
Diffraction .....	246
Psychoacoustics.....	248
The Ear.....	248
Pitch .....	249
Threshold shift .....	252
Masking .....	253
Acoustics effects .....	253
Timbral consonance .....	254
Digital Audio Effect.....	256
First-order shelving filters .....	257
Peak filters (or second order filters) .....	259
Integer and fractional delay filters – FIR Filters .....	260
Least-squares FIR .....	262
Windowing methods .....	264
GLS Methods .....	264
Lagrange Interpolation.....	265
Integer and fractional delay filters – IIR Filters.....	272
Thiran all-pole LP design.....	275
Delay-based effects – Comb filters.....	278
Universal comb filter .....	281
Parallel connection of comb filters .....	281
Delay-based audio effects.....	281
The Hammond Organ .....	282
Modulators.....	283
Amplitude Modulation .....	283
Frequency modulation .....	283
Recap on maths .....	285
Fourier Transform .....	285
Multidimensional Fourier Transform .....	285
Fourier Series .....	286
Acoustic Field .....	286
Differential operator .....	286
Gradient Operator .....	286
Divergence Operator.....	286
Laplace Operator .....	287
Homogenous wave equation and homogeneous Helmholtz equation.....	287
From Cartesian to spherical coordinates.....	290

Cylindrical coordinates.....	297
Inhomogeneous Wave Equation.....	300
Line Source.....	301
Boundary condition.....	302
The Kirchhoff-Helmholtz Integral .....	303
Plane wave representations .....	306
Spherical Harmonics Representations.....	307
Cylindrical Harmonics representations .....	310
Spatial Perception of Auditory Events.....	311
Primary Auditory Cues – Directional Cues in free-field .....	312
Duplex Theory .....	312
Monoaural Spectral cues .....	313
Primary Auditory Cues – Directional Cues in reverberant environments .....	314
Precedence effect .....	314
Distance estimation .....	315
Head-Related Transfer Function .....	315
Open Research topics .....	319
Signal Modelling: Signal based approach (Part 1) .....	320
Biquadratic Oscillator.....	324
Digital waveguide Oscillator .....	324
Dynamic amplitude control .....	324
Dynamic frequency control .....	325
Wavetable sinusoidal oscillator .....	325
Time segment-based models .....	326
Time compression/expansion .....	326
Granular Synthesis .....	328
Concatenative synthesis .....	329
Spectral models: sinusoidal modelling of deterministic signals .....	329
Implementation .....	330
Comparison between fft-based approach and time-domain approach.....	331
Synthesis-by-Analysis .....	331
Additive model .....	331
Wave Synthesises .....	333
Discrete-Time Sawtooth Oscillator.....	333
Square wave oscillator .....	336
Triangle wave .....	336
Source-Filters Models.....	337
Nonlinear modelling .....	339
Wave shaping .....	339
Overdrive and distortion as musical effects .....	340
Asymmetric distortion .....	340
Multiplicative synthesis.....	341
Amplitude modulation .....	342
Synthesis by frequency modulation .....	342
Compound modulation.....	343
Nested modulation .....	343
MATLAB .....	344
Modulation on Matlab.....	344
Linear predictive coding on Matlab .....	346

Source-Based Methods .....	348
Origins .....	348
Physical Model Synthesis (PMS) .....	348
Yamaha .....	349
Korg .....	350
Technics .....	350
Seer Systems .....	350
Cakewalk .....	350
Generalmusic .....	350
A quick recap from physics .....	351
Kirchhoff variables .....	353
Modal decomposition .....	353
Functional blocks .....	356
Feed-forward vs. feedback .....	357
The Karplus-Strong algorithm .....	357
General Least Squares (GLS) Method .....	361
Maximally-Flat Fractional Delay Filter .....	361
All-pass fractional delay filters .....	365
Discretisation of Lumped models .....	367
Courant-Friedrichs-Lowy condition .....	369
FD approximation of a lossy string .....	371
Stiffness in FS strings .....	372
Example .....	372
Modal synthesis .....	375
DT mechanical oscillators .....	378
Modal analysis .....	379
Modal synthesis .....	379
From Digital WaveGuides to WaveGuide Networks .....	380
The string .....	380
Acoustic Tube .....	384
Kelly-Lochbaum junction .....	386
N-dimensional and loaded junctions .....	387
Wave Digital Structure .....	388
Wave Digital Filters principles .....	389
Resistor .....	389
Generic impedance .....	389
Capacitor .....	390
Inductor .....	390
Voltage Generator .....	391
Connecting two dipoles .....	391
Adaptors .....	392
Series Junction .....	392
Parallel junction .....	394
Modelling an Analogue circuit .....	396
Modelling NL elements in the WD domain .....	397
Example: Chua's circuit .....	399
Modelling and implementation of Wave Digital Filters .....	401
Circuit Theory .....	401
Definition of voltage waves .....	402
Linear resistor .....	403
Linear resistive voltage generator .....	403
Linear Dynamic Elements .....	404

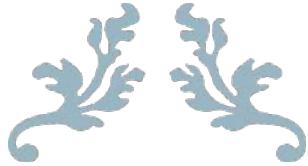
Linear Capacitor .....	405
Linear Inductor .....	405
Linear Wave Digital One-Port Element .....	406
Nonlinear elements .....	407
Modelling the topology .....	408
WD Junctions (Adaptors) .....	410
Connection tree Structures .....	411
Computational flow in Connection Trees .....	413
MATLAB implementation .....	417
Spatial Sound with Headphones: Binaural Rendering .....	432
Binaural recordings .....	432
HRTF-Bases Binaural Rendering .....	433
BRS with room model .....	434
Motion-Tracked Binaural .....	435
Spatial Sound with Loudspeakers - Stereophony and Panning .....	437
Stereophony .....	438
Channel based .....	438
Transform-domain based .....	439
Two-channel stereophony .....	439
Model based stereophony .....	439
Channel based stereophony .....	442
Multichannel stereophony .....	446
Model based stereophony .....	446
Channel based stereophony .....	448
Huygens' principle .....	449
Kirchhoff Helmholtz integral .....	450
Derivation of wave field synthesis – 3D Wave field Synthesis .....	453
Derivation of wave field synthesis – 2D Wave field Synthesis .....	454
Derivation of wave field synthesis – 2.5D Wave field Synthesis .....	455
Artefacts of wave field synthesis .....	458
Wave Field Analysis .....	459
Wave field Extrapolation .....	459
Ambisonics .....	460
Planar Geometry (2D Case) .....	460
Plane wave source .....	462
Line source .....	464
Data-based scenario .....	464
Rendering Approaches .....	465
Loudspeakers modelled as plane waves .....	465
Loudspeakers modelled as line sources .....	470
3D Geometry system .....	475
Model based-scenario .....	476
Data-based scenario: first-order ambisonics .....	477
Data-based scenario: HOA .....	477
Data-based scenario in general .....	478
Recap on the differences between wavefield synthesis and ambisonics .....	480

## IX

Reverberation Algorithms .....	481
History: .....	481
Impulse response:.....	481
Descriptors of reverberation .....	482
The Arvedi Auditorium in Cremona.....	484
The reverb problem .....	485
Reverberation algorithms .....	486
Early reflections + Late Reverb.....	487
Early reflections.....	487
Late Reverberations .....	487
Criteria for the realism of a reverberation algorithm .....	490
Schröder's reverberator.....	491
JCRev .....	491
Feedback Delay Networks.....	492
Geometric reverberation Algorithms .....	495
Finite Difference Methods .....	495
Boundary Element Method.....	495
Image Source Method.....	495
Beam tracing .....	497
Summary on geometric reverberation algorithms.....	499
Visibility region .....	499







---

# DIGITAL AUDIO ANALYSIS AND PROCESSING

---

First Module



A.Y. 2023/2024  
POLITECNICO DI MILANO  
MUSIC AND ACOUSTIC ENGINEERING



## Introduction to the course

He starts saying that this is a weird year since most of the courses that were held by Augusto Sarti are now covered by other teachers and they are still trying to find the right balance.

### Topics:

- **Module 1:** this module will be focused mostly on analysis and, to a small extent, on applications. He gives for granted the basis acquired in CMRM. We start by studying the basics of the analysis: we will ask ourselves how can we analyse a signal that can be assumed as a sum of sinusoids. We will also introduce the aspect of filter banks. Then we move to sound processing tools and adaptive filtering. We will conclude with a part on space-time processing, feature extraction and analysis. We do not cover anything that touches deep-learning and machine-learning. Just a smattering of the fundamentals. Which will the applications be? Pitch tracking, vocoder, envelope tracking, time warping, time and pitch scaling, pitch tracking. Echo cancellation, feedback cancellation, noise reduction, etc...; beamforming (changing the direction to acquire sound), acoustic source localisation and extraction.




---

*A smattering of Un'infarinatura su*

---

### References:

Only the slides.

- **Module 2 (SSSP):** the second module will deepen some aspects about musical signal modelling. We will learn different ways to synthetise sound: additive spectral modelling, space-time modelling, modal modelling, physical modelling (electrical circuits), digital waveguide networks, wave digital structures. Then, we will study all about Reverberation. A rough idea to understand what reverberation is, could be the following: we do not synthetise the sound from scratch, but we just change an acquired signal to obtain another signal that could be as close as possible to the one radiated by the actual source. Then we will move to Spatial Audio. To solve the problems related to that topic we will have to keep two things in mind: the former that we are in a 3d environment, and we have to get the sound by respecting the effect of the environment itself. The latter is that we have to recreate a sensation of that environment by using just two channels (L and R).

### References:

Only the slides.

### Evaluation:

The two modules are valued separately. For each module there will be one written test. In addition to that, there will be a homework assignment for each module that consists of a problem to be solved at home in pair. (They are not actually sure if it will be whether in pair or not, it will depend on how much complex the homework is. The final score will be evaluated with a weighted average between the written test (80%) and the homework (20%). The rules for the HW are the same by Giampiccolo (+3 points bonus if we do not go over the first deadline, 0 points bonus if we submit it within the 14<sup>th</sup> July (French Revolution), and -5 point malus if we go over it). With the homework, we have to present also a report or a presentation.

They are thinking to remove the mid-term exam.

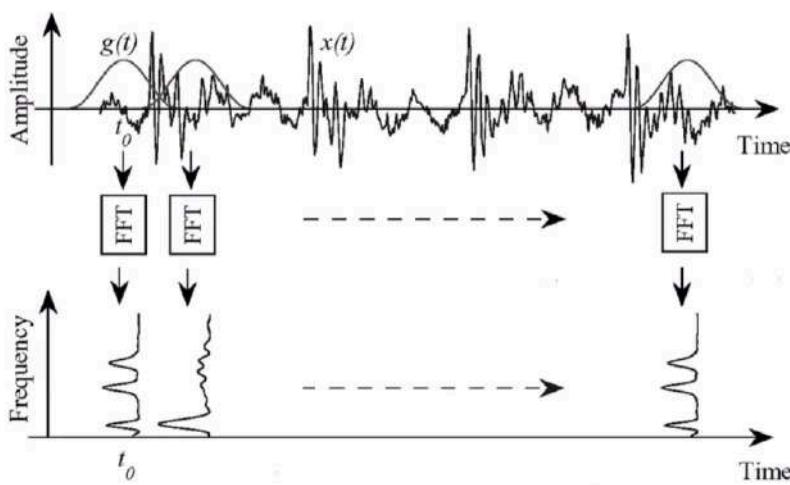
## Sound Analysis

During this series of lectures, we will learn how to extract information from an audio signal that can be seen as a series of sinusoids. This question makes sense just when we connect it to the sound analysis.

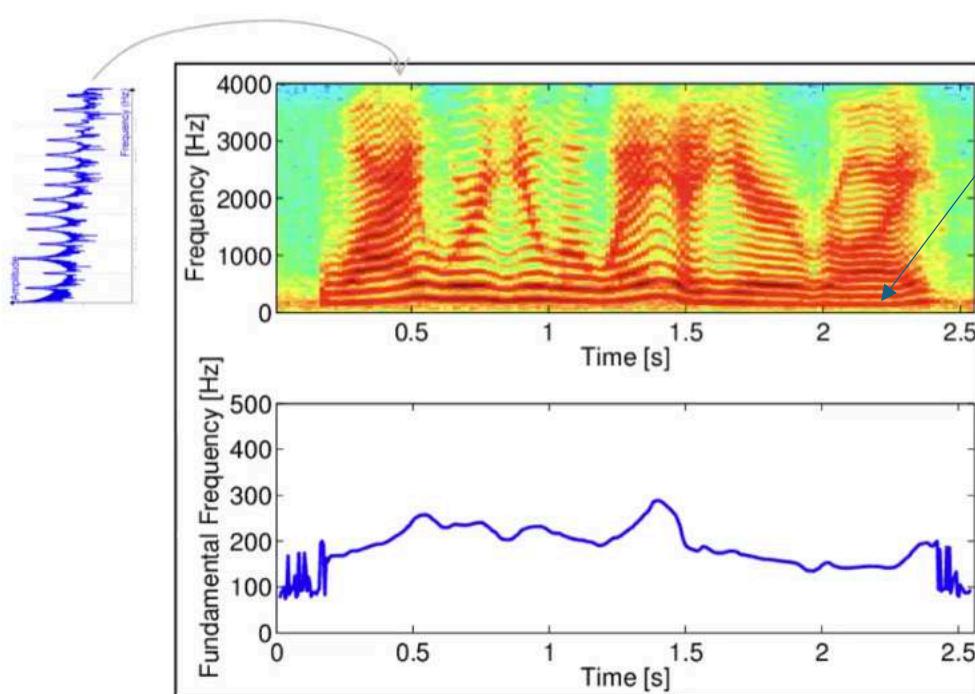
First of all, why do we need to extract sinusoids from a signal?

We can answer to this question with an example about autotune: let us suppose that we want to correct the pitch of a singer from a mastered song. We cannot simply change the pitch of the whole song. In an ideal case, we would have to previously extract the singer part, then change the pitch of that part and, at the end, remaster everything together. But it is not as easy to extract a single part from a mixed track. So, how can we achieve this result without extracting the part? By exploiting the Fourier Series: we will just have to correct the sinusoids related to the frequencies covered by the singer!

### Short Time Fourier Transform

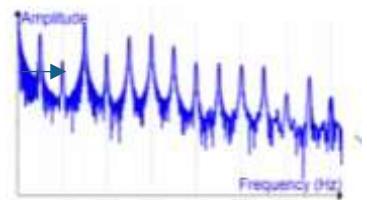


One of the most powerful tools to extract sinusoids is the so-called STFT (Short Time Fourier Transform). As we can see in the picture on the left, by applying the STFT we take a certain portion of the signal and we evaluate its DFT by means of the FFT. Then we stack the single DFTs one next to the other obtaining the so-called spectrogram.

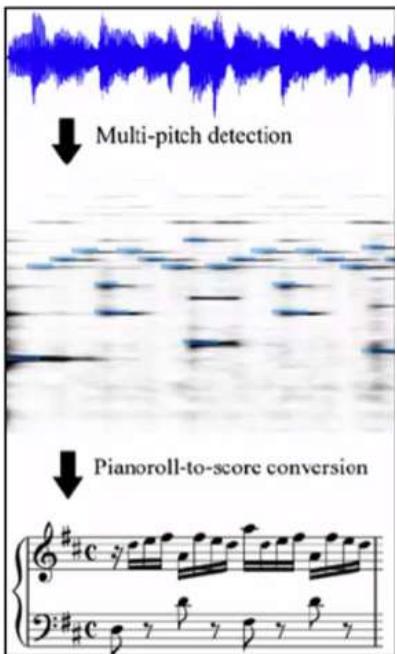


Out of curiosity: the spectrogram in the picture represents the typical spectrum of a speech signal.

Let us see how to extract some pieces of information from the spectrogram. For instance, how can I find the pitch of the speech? I have to look for the fundamental frequency of the signal. How can I find it? As the distance between all the main peaks of the spectra.



From pitch tracking, we can also move to automatic music transcription. Let us see a typical pipeline.



1. First, we break down the signal into a series of frames;
2. Then, we compute the STFT;
3. After that, we transform the STFT into the piano roll (that can be seen as nothing but a spectrogram with tones instead of the frequencies);
4. We transcribe the spectrogram in a sheet music.

**Out of curiosity:** in forensic analysis we use the fluctuations of the frequency of the current (50Hz/60 Hz) to identify which is the position in time and space of a certain fact. We know that the frequency is not always stable at 50 Hz (or 60 Hz in the USA). It often happens that its value floats around that value. When we want to understand when a certain fact happened in time, we just have to compare the interferences related to the power supply present in the signal we are analysing with the history of the frequency fluctuation registered by the society that provides energy (ENEL).

Another field in which this kind of analysis (STFT) is used is the beamforming that is nothing but the detection of the direction of arrival of a specific sound wave in space.

After having the signal analysed, we can resynthesise the sound.

We can apply the concepts of audio compression but also of transformation techniques for music synthesis and sound composition.

This interesting table reports how difficult it is to use some techniques instead of others to obtain a sound that has been analysed with a certain technique,

Analysis ⇒ Synthesis ↓	STFT	Filterbank
I-STFT	Easy	Hard
Bank of Oscillators	Hard	Easy

As we can see, the easiest way to synthetise a signal is either by inverting an STFT or by using a bank of oscillators exploiting the information coming from an analysis made by means of filter banks.

We can also mix the two worlds but we may encounter some challenges.

So, the overall pipeline of an analysis-plus-synthesis scheme typically is

1. Analysis (in which we determine the sinusoidal components) performed by
  - a. STFT (that favours frequency domain over time domain)
  - b. Filter Bank (that favours time domain over frequency domain)
2. Data reduction (optional - if we want to optimise space by using special algorithms (MPEG1, etc...))
3. Modification (optional – autotune, filtering, etc...)
4. Synthesis
  - a. Inverse STFT
  - b. Bank of Oscillators

At the end of this module, we should be able to implement applications that follows this pipeline.

### Resolving sinusoidal peaks using STFT

Setup:

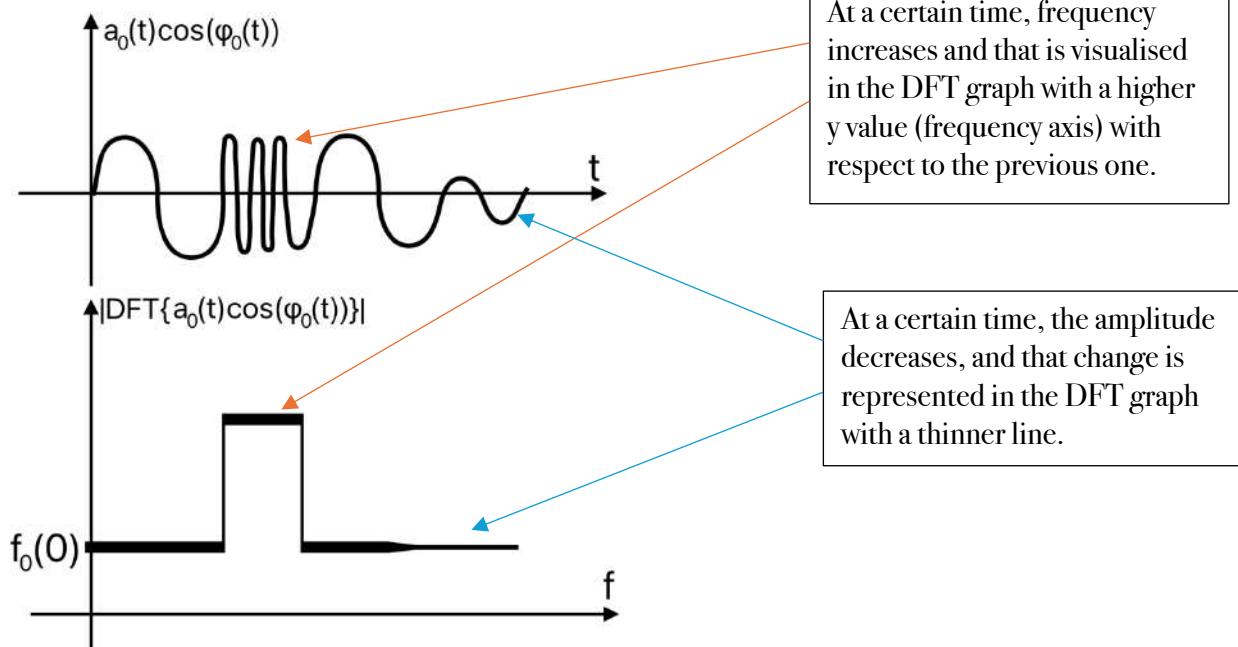
The signal under analysis can be well approximated by a sum of sinusoids

$$s_s(t) = \sum_k a_k(t) \cos(\phi_k(t))$$

Where:

$$\phi_{k(t)} = 2\pi f_k(t) + \varphi_k$$

How can we interpretate the argument of the sum? Frequency can change in time but also amplitude can change in time. Let us visualise it with a graphical example.



Our **goal** is to estimate some parameters:

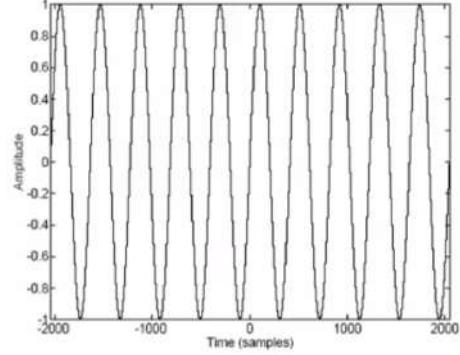
- Number of sinusoids
- Frequency
- Amplitude
- Phase

The problems come from the fact that the signals that we are analysing are not ideal. Most of all, because they do not have an infinite length (fundamental to obtain a finite description in frequency). Furthermore, the signals are windowed! The analysis is often performed frame by frame so the length in time is surely not infinite.

Let us see what happens when we evaluate the DTFT of an ideal signal (infinite and without windowing).

Let us consider a complex sinusoid:

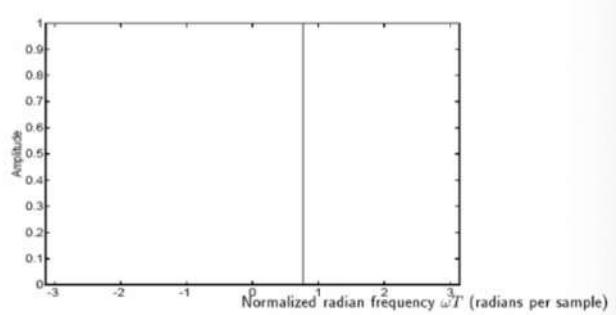
$$x(n) = e^{j\omega_0 nT}$$



Its Fourier transform will be equal to

$$X(\omega) = \delta(\omega - \omega_0)$$

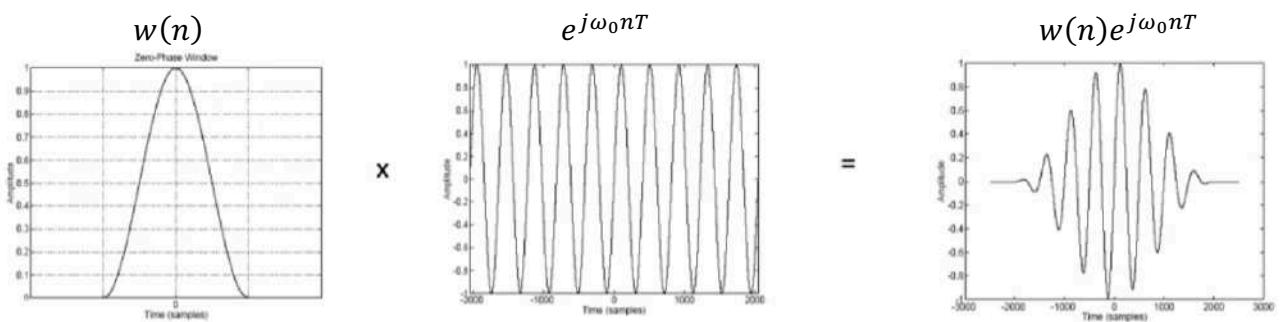
So, the spectral representation of a complex exponential is a Dirac delta at the frequency  $\omega_0$ .



Out of curiosity: we prefer to represent sinusoidal signal with the exponential notation even though their imaginative part is equal to zero because of this simple representation they have in the frequency domain. We just have to remind that when we go back into the real world, we have to extract the real part from the complex signal.

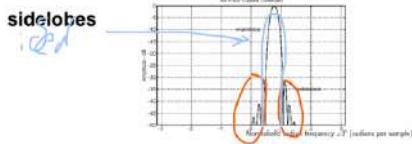
When we window a signal, we multiply point by point the signal to the window. In this way, we remove some components in time domain changing the signal. It is no longer a pure sinusoid!

By considering a window expressed by  $w(n)$ , we obtain a figure like the one reported in the graph below.



Multiplication in time domain means convolution in frequency domain. This operation generates some tails that are plotted as sidelobes in the function representation.

Trying to take notes as fast as I could during lecture, I tried to highlight the sidelobes and the main lobe in the following picture obtaining a peculiar result.

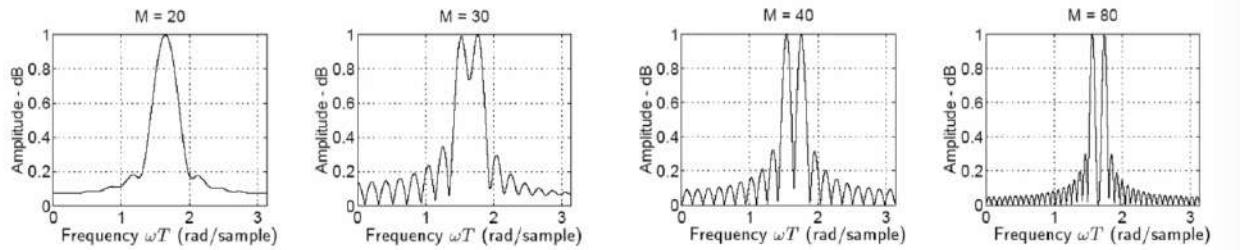


Going back to the example of the voiced signal, to detect the pitch of this portion, we just have to identify the point of maximum of the function, so the frequency around which the light blue lobe is centred.

The problem comes when we have more than one sinusoid. In fact, by adding more sinusoids it becomes difficult to:

- Detect the **number of sinusoids/peaks**.
- Detect the precise **frequency** of each sinusoid.

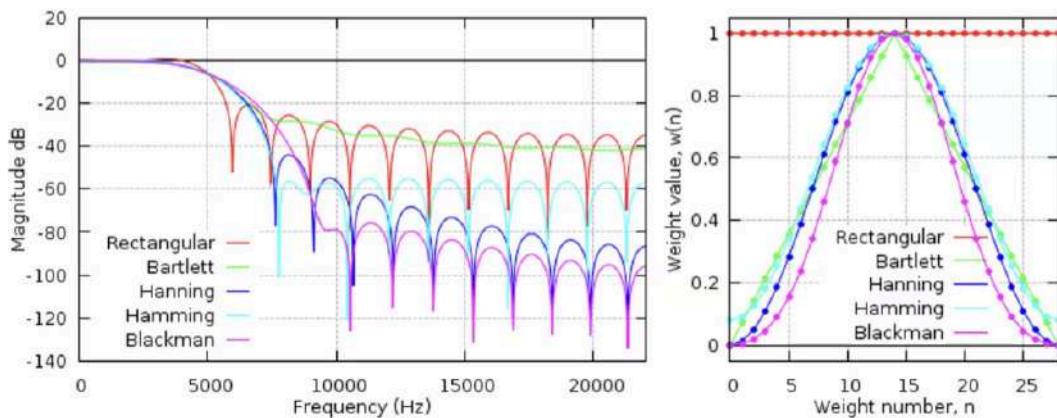
Let us make it clear looking how the number of samples ( $M$ ) affects the DFTF of a two-cosines signal windowed with a rectangle function.



As we can notice, the functions overlap one with the other. Why are things changing with  $M$ ? Because the resolution changes with how long the window is.

Not just the length makes differences but also the type of window. Different windows have different properties:

- Main Lobe width
- First sidelobe height



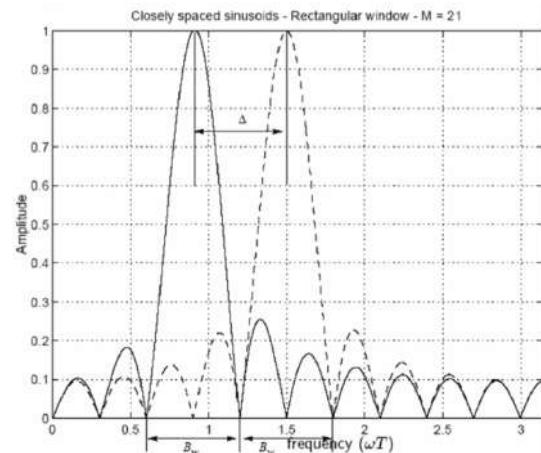
Simply speaking, the differences among these windows are the amplitude of the main lobe and the “speed” with which the window goes down. We report a table with the properties of the main windows.

Window Type	Peak Sidelobe Amplitude (dB)	Approximal width of Main Lobe	Roll off
Rectangular	-13	$\frac{4\pi}{M-1} \approx 2\Omega_M$	6 dB/oct
Barlett (triangular)	-25	$\frac{8\pi}{M} = 4\Omega_M$	12 dB/oct
Haan	-31	$\frac{8\pi}{M} = 4\Omega_M$	18 dB/oct
Hamming	-41	$\frac{8\pi}{M} = 4\Omega_M$	6 dB/oct
Blackman	-58	$\frac{12\pi}{M} = 6\Omega_M$	18 dB/oct

So, how do we choose the window length?

Let us find which is the situation in which we are happy.

- A conservative requirement for resolving 2 sinusoids (in noisy conditions) with a spacing of  $\Delta Hz$  is to choose a window length long enough so that their main lobes are clearly discernible. For example, we may require that their main lobes meet at the first zero crossings in the worst case (at the narrowest frequency separation).
- To obtain the separation shown on the right, we must have  $B_W \leq \Delta$ , where  $B_W$  is the main lobe width, and  $\Delta$  is the minimum sinusoidal frequency separation.
- $B_W$  can be expressed as  $B_W = L \frac{F_s}{M}$ , where:
  - Rectangular window  $\rightarrow L = 2$
  - Hamming window  $\rightarrow L = 4$
  - Blackman window  $\rightarrow L = 6$



In other words, we are happy if in the worst-case scenario, the distance between the two closest sinusoids is not smaller than the bandwidth of the main lobe that I am using. We are delighted if there is no interaction or overlapping between the two lobes.

If  $\Delta = B_W$ , the two minima of the lobes touch one with each other. So, the constrain in the worst-case scenario to be applied will be  $B_W = \Delta$ . The  $B_W$  of a window is equal to  $B_W = L \frac{F_s}{M}$  where  $L$  is a number related to the window (parameter peculiar of the kind of window),  $F_s$  is the sampling window and  $M$  is the number of samples.

So, the constrain that will have to respect it

$$B_W = L \frac{F_s}{M} \rightarrow M \geq L \frac{F_s}{\Delta} = L \frac{F_s}{f_2 - f_1}$$

What does it tell me? I made an assumption that there is a certain  $\Delta$  between the two closest frequency of the signal, then I choose the window I want (so I fix  $L$ ) and, at the end, I have two parameters to set: the sampling frequency and the length of the window.

The  $\Delta$  is directly linked to the concept of **frequency resolution**: we can define *resolution* as the minimum number of samples  $M$  that allows us to distinguish the two sinusoids with the closest frequency in a certain signal. If  $M$  is greater than or equal to the resolution, we would be able to see two separated peaks that are not merged in any way.

## Resolution

There are special kind of signals that have some peculiarities that make them remarkable:

- Signals from tonal musical instruments (harmonic signals): they are characterised by a series of sinusoids spaced from each other by a constant value equal to the fundamental frequency.
- Speech signals: the spectra of this particular kind of signal are characterised by the presence of formants.

So, let us consider a harmonic signal, with a fundamental frequency  $f_0$ . We will have that all the harmonic components will occur at integer multiples of  $f_0$ , they will be spaced by  $\Delta = f_0$  and, in order to make them distinguishable, we will have to impose that:

$$M \geq L \frac{F_s}{\Delta} = L \frac{F_s}{f_0}$$

This means resolve a signal: being able to distinguish all its frequency component.

What is the interpretation of this?

We notice that  $\frac{F_s}{f_0} = \frac{\omega}{T_s} = P$ . So  $P$  represents the number of samples that I have in one period sinusoid.  

$$\begin{aligned} f_0 &= \frac{1}{T_0} \\ F_s &= \frac{1}{T_s} \end{aligned}$$

Hence our resolution criterion becomes:

$$M \geq LP$$

By considering the windows seen in the previous lectures, to fully resolve the harmonics of a periodic signal with period  $P$  samples, we need at least:

- 2 periods under the rectangular window
- 4 periods under the Hamming window
- 6 periods under the Blackman window

If we want to abstract a general law, we can tell that within one frame (my window) I should be able to observe  $L$  period of the sinusoids associated to the fundamental frequency.

Let us see it with an example:

We want to resolve peaks of two singers considering the following setup:

- Male ( $f_0 \geq 100$  Hz) and female ( $f_0 \geq 200$  Hz) singers. *As we can notice, we need to know the lower boundary of their vocal range.*
- Sampling frequency  $F_s = 50$  KHz

Depending on the window used, we need at least M samples:

	MALE $f_0 = 100 \text{ Hz}$	FEMALE $f_0 = 200 \text{ Hz}$
RECTANGULAR $L = 2$	$M \geq L \frac{F_s}{f_0} = 1000$	$M \geq L \frac{F_s}{f_0} = 500$
HAMMING, HANNING $L = 4$	$M \geq L \frac{F_s}{f_0} = 2000$	$M \geq L \frac{F_s}{f_0} = 1000$
BLACKMAN $L = 6$	$M \geq L \frac{F_s}{f_0} = 3000$	$M \geq L \frac{F_s}{f_0} = 1500$

So, what have we learnt?

Given a signal approximated as sum of sinusoids, knowing the minimum distance between possible frequency peaks  $\Delta$  (desired frequency resolution equal to the fundamental frequency  $f_0$  if the signal is harmonic) and given a window function  $L$ , we are able to tell how long the window should be to resolve the signal.

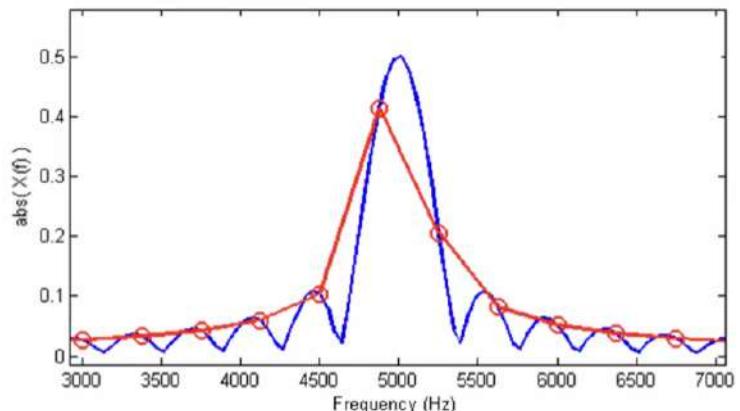
$$M \geq L \frac{F_s}{\Delta} \underset{\substack{\omega \\ \text{Harmonic}}}{=} L \frac{F_s}{f_0} \underset{\substack{\omega \\ \text{Signal}}}{=}$$

### Spectral interpolation

The teacher has hidden something to our eyes during all this time: he always showed continuous transformants. But we deal with digital signals! The transformant of a periodic discrete signal is, in turn, discrete. A plausible representation of the DFT can be a series of points like the red ones in the picture on the right. How can we find the maximum of the DFT? It is not that easy because the sample with the higher value does not necessarily need to be in correspondence of the maximum of the blue curve (we remind that we can see a DFT as the sampled version of a DTFT).

How can we cope with it? There are two possible solutions:

1. Ideal interpolation (**zero padding** in the time domain)
2. **Parabolic interpolation** (possibly in addition to zero padding)



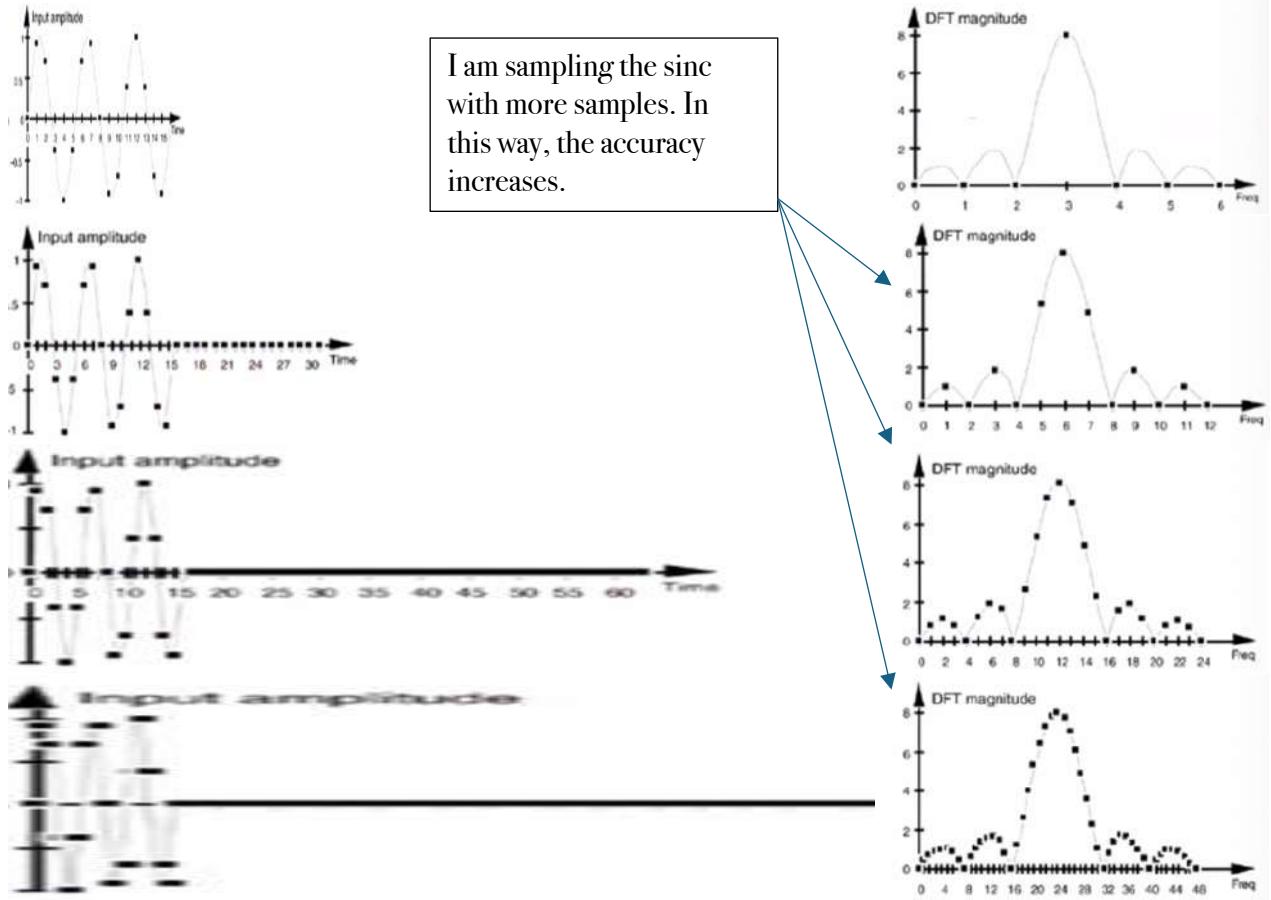
Let us analyse them!

#### Zero Padding

It is the smartest solution, it costs nothing! Which is the idea under it?

We add some zeros at the end of the signal in order to increase the number of samples. In this way the

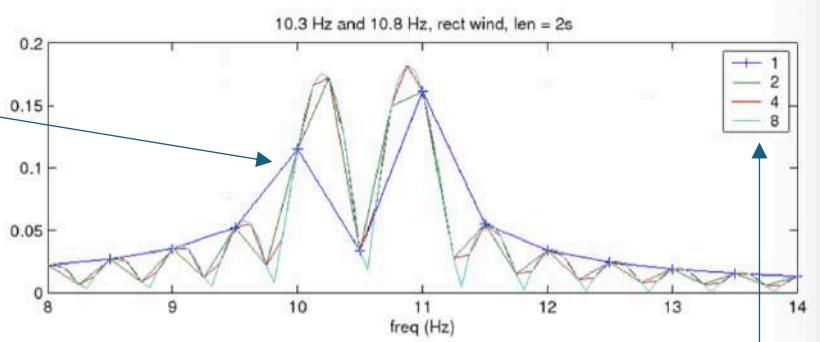
magnitude of the DFT remains the same (since the contribution of the zero is always a zero  $0 \cdot e^{-j2\pi n \frac{f}{F_s}} = 0$ ), but we increase its resolution given the fact that there are more points to represent it. In fact, we remind that a zero padding in time domain corresponds to an interpolation in frequency domain.



For sinusoidal peak-finding, spectral interpolation via zero-padding gets us closer to the true maximum of the main lobe when we simply take the maximum-magnitude FFT-bin as our estimate.

Let us see in the following images how peak finding changes with various oversampling factor (obtained by zero padding).

We move from the blue curve to the turquoise one.



The numbers point out how many zeroes we add to the end of the signal. The formula is the following. By considering a signal of length  $N$ , the length of the new signal will be:

$$N * \text{number}$$

So in the first case (1) we add no zeros;

In the second case we add  $N$  zeros ( $N + N = 2N$ )

In the third case we add  $3N$  zeros ( $N + N + N + N = 4n$ )

We can notice how the two lobes are far more precise in the turquoise case rather than in the blue one.

First of all, we notice that the peak localisation error is half the distance between consecutive frequency samples. But, wait a minute, why half the distance?



If we evaluate how many “spaces” we have from 0 to  $F_s$  we have simply to divide  $F_s$  by the number of samples. This number represents also the distance between one sample and the other, distance that we would call error. But that is wrong!

Actually, each samples collects not all the frequencies that precedes its frequency but just the frequencies that lies in a bandwidth that goes from its frequency  $-1/2$  and its frequency  $+1/2$ .



So, the peak localisation error will be equal to

$$\Delta f = \pm \frac{F_s}{2N}$$

In the previous example, by localising the fundamental frequency of the male singer using a FFT of length 1024, the localisation error could have been of  $\pm \frac{F_s}{2N} = \pm \frac{50\text{ KHz}}{2 \cdot 1024} \cong \pm 25\text{ Hz}$  that is the 25% of the fundamental frequency ( $f_0 = 100\text{ Hz}$ ). Is it much or little?

To answer to this question, we have to recall the concept of JND (Just Noticeable Difference). The JND is the minimum frequency change in a pure tone (sinusoid) that we are able to perceive. For a normal man, the JND value is:

$$JND = \begin{cases} 0.5\%f, & f \geq 600\text{ Hz} \\ 3\text{ Hz}, & f < 600\text{ Hz} \end{cases}$$

So, a 25 Hz frequency sliding is extremely perceivable.

To solve this problem, we need to zero pad the sequence, extending it from the length  $M$  if the window do a much longer length  $N$  of the FFT. How long has to be the FFT? It must be long enough to have  $\Delta f \leq JND$ . So, by considering that  $f_0 = 100\text{ Hz} < 600\text{ Hz}$ , we must have  $N \geq \frac{F_s}{2 \cdot JND} = \frac{50\text{ KHz}}{2 \cdot 3\text{ Hz}} \cong 8333$ . This means that we have to oversample of factor greater than eight.

In general, zero padding a windows signal corresponds to perform spectral interpolation. That of spectral interpolation is, in fact, a general way to approach the problem of localisation accuracy.

**Quick remark:** what if rather than using an oversampling of zeropadding of a factor 8, I decided to enlarge the window? Sometimes it works, sometimes not because as we increase the resolution in frequency domain, we lose temporal resolution at the same time. Enlarging the window could become useful as long as we work with slow signals. If we are analysing the sound of an instrument that is played very fast, we need high resolution in time, so we cannot use this method.

### Parabolic Interpolation

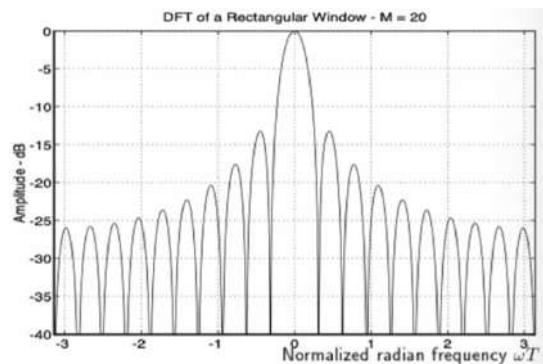
With a parabolic interpolation we fix two points, and we find which is the parabola that interpolates them.

Why a parabola? Because the sinc recalls the parabola and also because with a Gaussian window

$(\frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{(t-\mu)^2}{2\sigma^2}})$ , we have that the transform of a Gaussian is always a Gaussian  $\frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{(f-\mu)^2}{2\sigma^2}}$ .

If we plot that gaussian in the frequency domain in dB, we apply the logarithm to an exponential log\_10

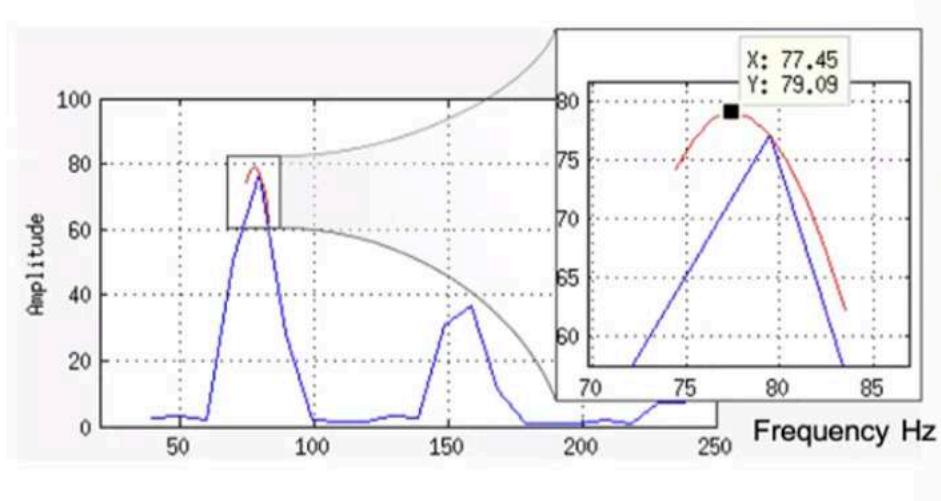
$\frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{(f-\mu)^2}{2\sigma^2}}$  obtaining an  $x^2$  so a Parabola.



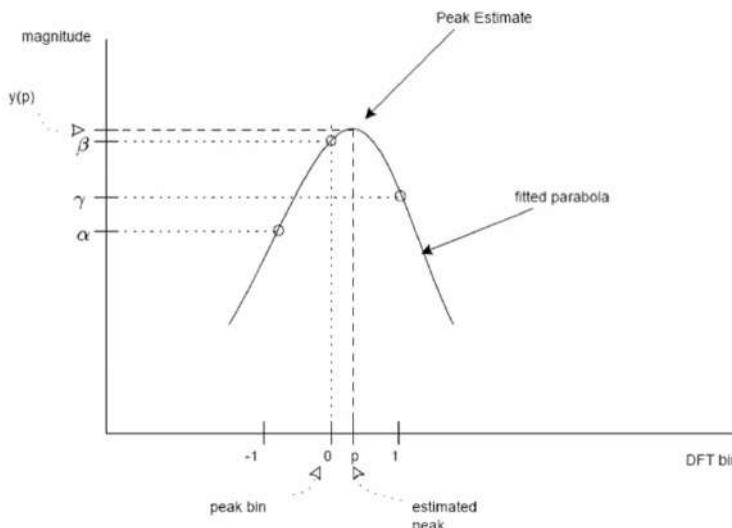
So, quadratic interpolation is completely precise when we are interpolating the spectral sample of a Gaussian window on a dB scale. In all other cases, the main lobe is just well approximated by an upside-down parabola.

Let us recall how the parabolic interpolation works:

We assume that the peak of a lobe could be approximated with a parabola.



Let us try to understand how to find the peak position by recalling what a parabolic interpolation is.



We assume the parabola to be “centred” at  $p$ , which is the parameter that we are trying to compute. This is expressed by the following formula:

$$y(x) \triangleq a(x - p)^2 + b$$

We plug into the formula the three points we want to interpolate, and we put the expression equals to a constant.

$$y(-1) = ap^2 + 2ap + a + b = \alpha$$

$$y(0) = ap^2 + b = \beta$$

$$y(1) = ap^2 - 2ap + a + b = \gamma$$

Now we put the three equations in a system, and we solve it:

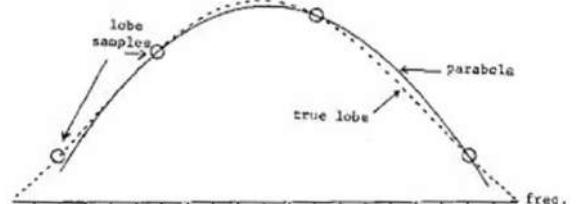
$$\begin{aligned}
 & \begin{cases} \alpha = ap^2 + 2ap + a + b \\ \beta = ap^2 + b \\ \gamma = ap^2 - 2ap + a + b \end{cases} \xrightarrow{\alpha-\gamma} \begin{cases} \alpha = ap^2 + 2ap + a + b \\ \beta = ap^2 + b \\ \alpha - \gamma = ap^2 + 2ap + a + b - ap^2 - 2ap + a + b \end{cases} \rightarrow \\
 & \rightarrow \begin{cases} \alpha = ap^2 + 2ap + a + b \\ \beta = ap^2 + b \\ \alpha - \gamma = 4ap \end{cases} \rightarrow \begin{cases} \alpha = ap^2 + 2ap + a + b \\ b = \beta - ap^2 \\ 2ap = \frac{\alpha - \gamma}{2} \end{cases} \rightarrow \\
 & \rightarrow \begin{cases} \alpha = ap^2 + \left(\frac{\alpha - \gamma}{2}\right) + a + (\beta - ap^2) \\ 2ap = \frac{\alpha - \gamma}{2} \end{cases} \rightarrow \begin{cases} \alpha = ap^2 + \frac{\alpha}{2} - \frac{\gamma}{2} + a + \beta - ap^2 \\ 2ap = \frac{\alpha - \gamma}{2} \end{cases} \rightarrow \\
 & \rightarrow \begin{cases} \frac{\alpha}{2} = -\frac{\gamma}{2} + a + \beta \\ 2ap = \frac{\alpha - \gamma}{2} \end{cases} \rightarrow \begin{cases} a = \frac{\alpha}{2} + \frac{\gamma}{2} - \beta = \frac{1}{2}(\alpha - 2\beta + \gamma) \\ 2ap = \frac{\alpha - \gamma}{2} \end{cases}
 \end{aligned}$$

At the end, we find  $p$

$$\begin{aligned}
 & \begin{cases} a = \frac{1}{2}(\alpha - 2\beta + \gamma) \\ 2ap = \frac{\alpha - \gamma}{2} \end{cases} \rightarrow \begin{cases} a = \frac{1}{2}(\alpha - 2\beta + \gamma) \\ p = \frac{\alpha - \gamma}{4a} \end{cases} \rightarrow p = \frac{\alpha - \gamma}{4 \cdot \frac{1}{2}(\alpha - 2\beta + \gamma)} \rightarrow \\
 & \rightarrow p = \frac{1}{2} \frac{\alpha - \gamma}{(\alpha - 2\beta + \gamma)} \in \left[-\frac{1}{2}, \frac{1}{2}\right]
 \end{aligned}$$

Remember that  $p$  is the peak location (in samples) from the largest sample. Using this, we have that the peak height estimate is

$$y(p) = \beta - \frac{1}{4}(\alpha - \gamma)p$$



The fact that we found a formula means that there is a close form solution to our problem.

Couple of notes. Up to this point we have not talked about the phase.  $p$  can also be used to interpolate unwrapped phase but phase interpolation is independent (can be any type and any shape). Since there is no reason to expect a “phase peak” where there is a magnitude peak, simple linear interpolation may be used (given a sufficiently large zero-padding factor). Matlab has an *unwrap* function, and there are quite a few papers on the subject in the IEEE Transactions on (Acoustics, Speech, and) Signal Processing.

Alternatively, the real and imaginary parts can be interpolated separately to yield a complex peak value estimate.




---

*yield cedere, far passare avanti*

## From analysis to synthesis

Let us see some techniques to synthetise sounds from information obtained by different analysis techniques.

### *Analysis with STFT and synthesis with Bank of Oscillators*

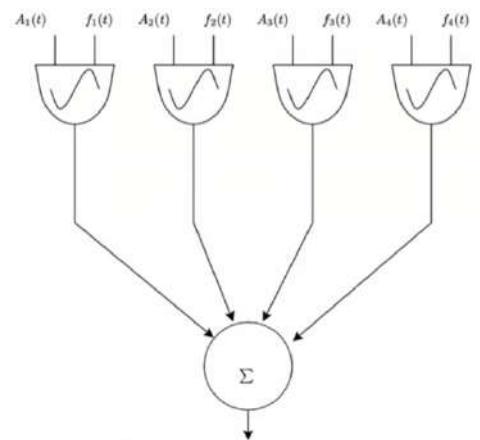
Let us see how to do it with a bank of oscillator (series of sinusoids). We bump into a problem of domain mismatch. Since we analyse a discrete signal, we do not have all the information we need to synthetise a continuous signal. This is why we pick information just in certain time instants and not for each frequency. This mismatch domain makes the synthesis of the signal very hard.

Analysis $\Rightarrow$ Synthesis $\downarrow$	STFT	Filterbank
I-STFT	Easy	Hard
Bank of Oscillators	Hard	Easy

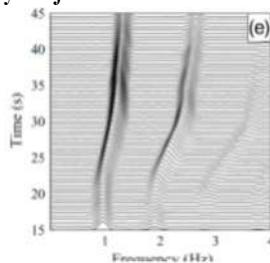
So, what are we doing:

Additive synthesis is a technique in which a signal is reconstructed from a summation of sinusoids. Each sinusoid has a time varying amplitude and frequency:

$$y(t) = \sum_{i=1}^N A_i(t) \sin[\omega_i(t)t + \phi_i(t)]$$



Let assume to have previously analysed the signal having obtained an amplitude and frequency trajectories for each sinusoidal component.

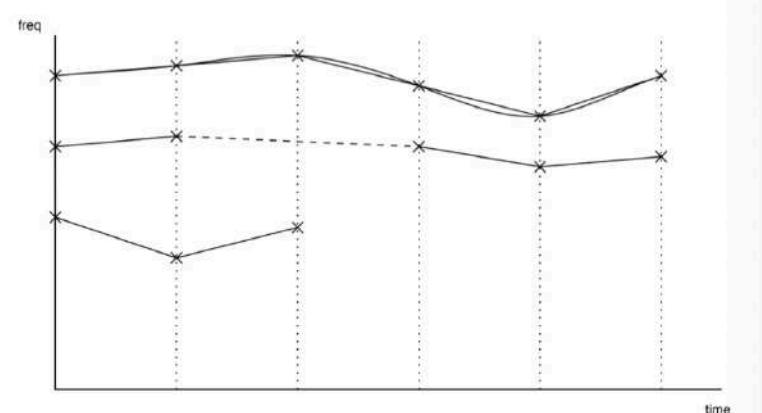
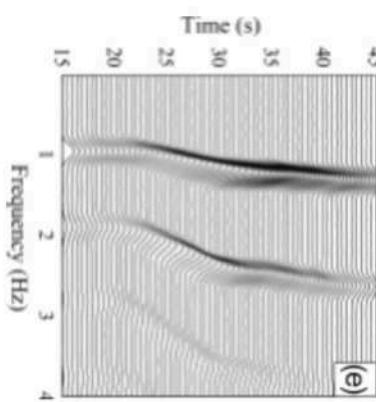


$$y(t) = \sum_{i=1}^4 A_i(t) \sin[\omega_i(t)t + \phi_i(t)]$$

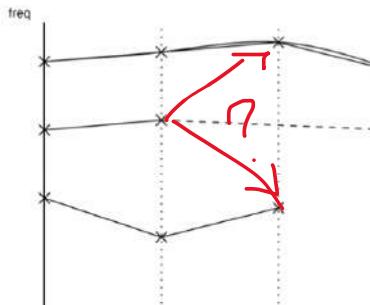
How can we find these trajectories from the discrete results we have?

It depends on whether we are interested in phase or not.

- Linear interpolation may be used to define the instantaneous amplitude and frequency between frames, when phase is discarded (this technique is called PARSHL);
- When phase is retained, cubic phase interpolation can be used from frame to frame (the technique developed by McAulay and Quatieri)



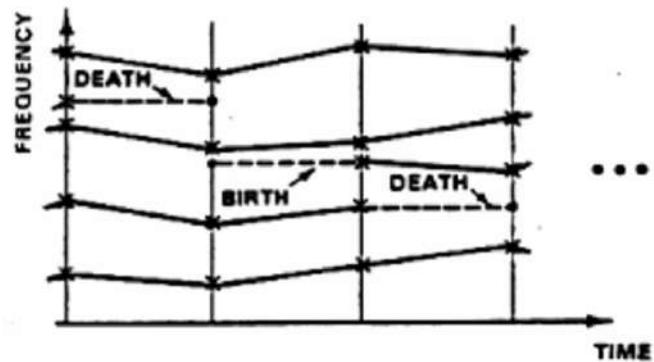
But still we have problems: we do not know which are the points that have to be connected together!



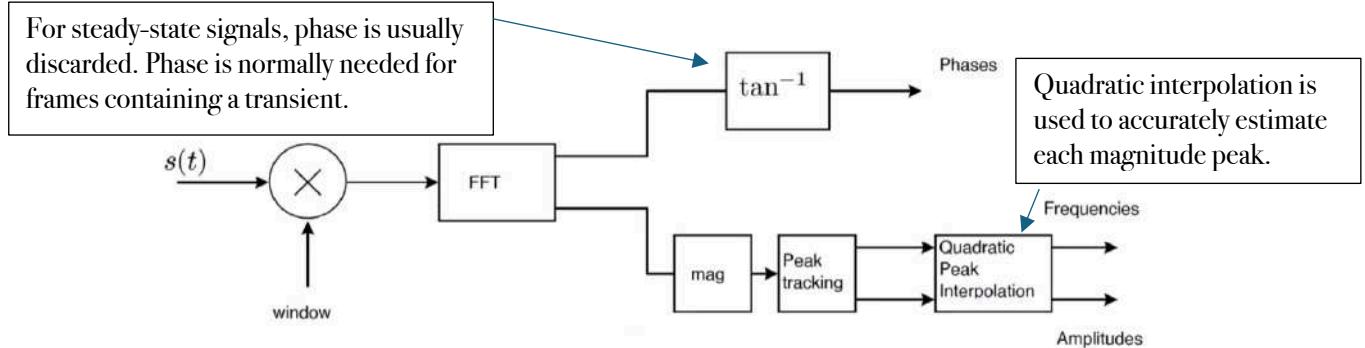
We do not have a unique solution to the problem.  
We have to make a choice. Try your best!  
The problem is that we want to identify topology from geometry.  
This is just a pompous way to say: "connect the dots".

There are some suggestions that can be followed:

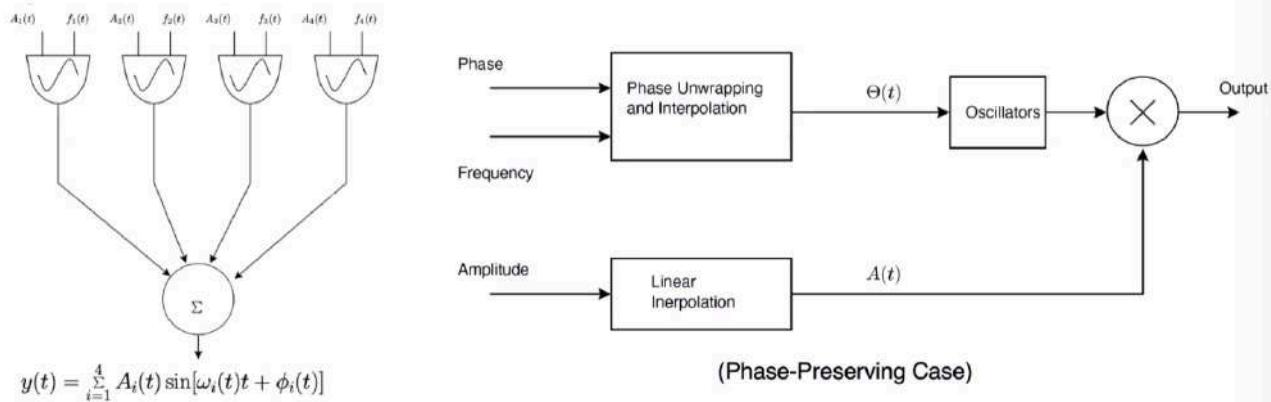
- Exploit harmonicity wherever possible (even when fundamental is absent)
- Do not rush into declaring a trajectory dead (in case there is a gap)
- Do not rush into declaring that a new trajectory is born (possible noise problems)



So, if the analysis scheme looks like the one reported there



The synthesis performed using a bank of amplitude and phase modulated oscillators:



We will talk about how to interpolate phases later on in the course.

But Maremma schiava! Not all the signals are harmonics! There are also noises. We must add a noisy component in our scheme. Let us see the model with *sines + noise modelling*

Spectral Modelling Synthesis (SMS) generalises the signal models used in PARSHL to include a filtered noise component:

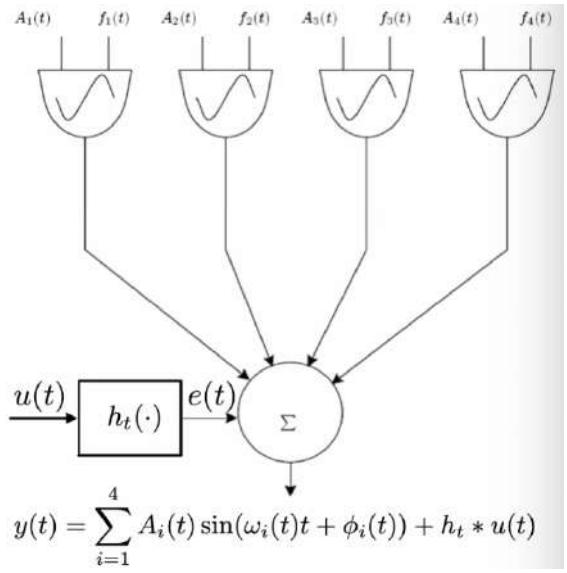
- The time-varying spectrum of the signal is assumed to be made up of a deterministic component (sinusoids) and a stochastic component (time-varying filtered noise)

$$s(t) = \sum_{r=1}^R A_r(t) \cos[\theta_r(t)] + e(t)$$

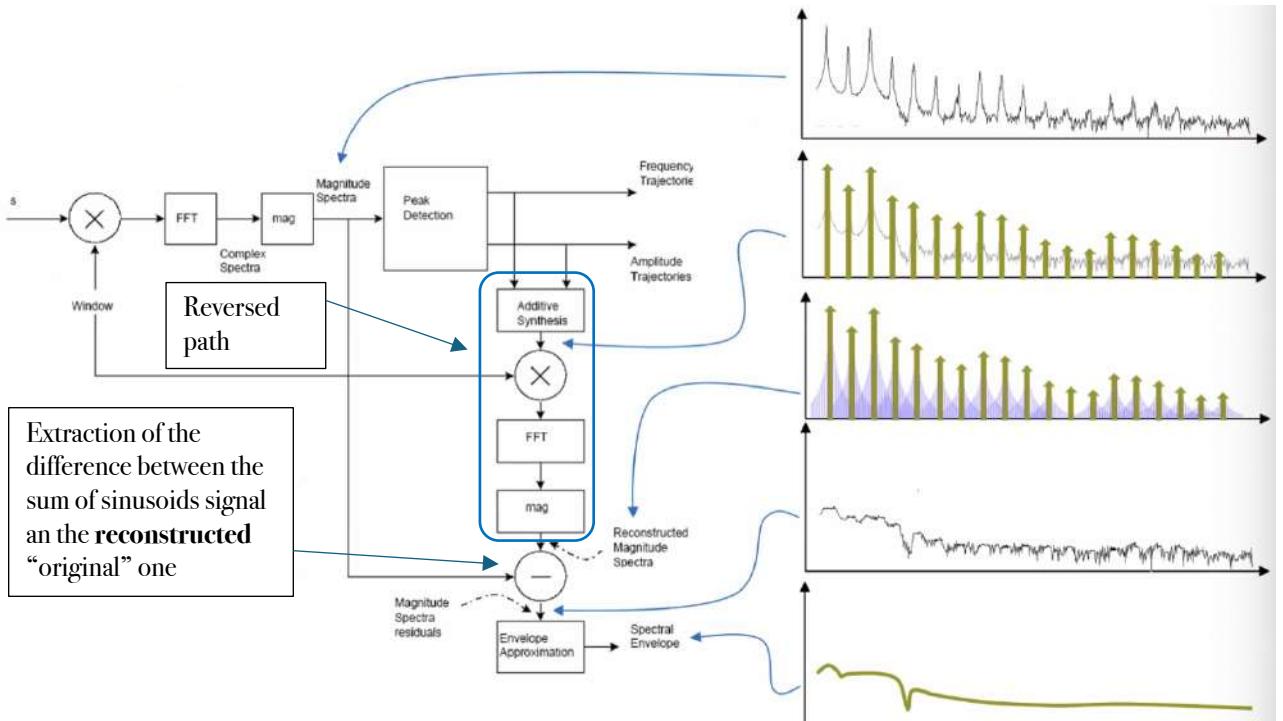
- Where  $A_r(t)$  and  $\theta_r(t)$  are the instantaneous amplitude and phase of the r-th sinusoidal component.  $e(t)$  is the residual, or noise signal, and is assumed to be filtered with noise:

$$e(t) = h_t * u(t) = \int_{-\infty}^{\infty} h_t(t - \tau) u(\tau) d\tau$$

- Where  $u(t)$  is the white noise, and  $h_t(\tau)$  is the impulse response of a time varying linear filter at time t. Specifically,  $h(\tau, t)$  is the response at time  $\tau$  to an impulse at time  $t$ . So, the noise used it is not random white noise: it is just the component of the signal that cannot be seen as a sum of sinusoids. Remind that the remaining part of the spectrum is estimated, and not the actual one.



How can we estimate the noise function? We have to change the analysis scheme.



### *Analysis with STFT and synthesis by ISTFT*

Another type of synthesis can be done by inverting the STFT. This procedure is so trivial that does not need explanations.

### *Analysis by using filter bank and synthesis by oscillators*

Another type of synthesis can be made by driving a bank of oscillators using filter banks data.

The signal domain is suitable for driving banks oscillators because it is not under-sampled over time. Historically this is the first analysis method, as it was available before computers exists (it was developed in late 1930s at Bell Laboratories).

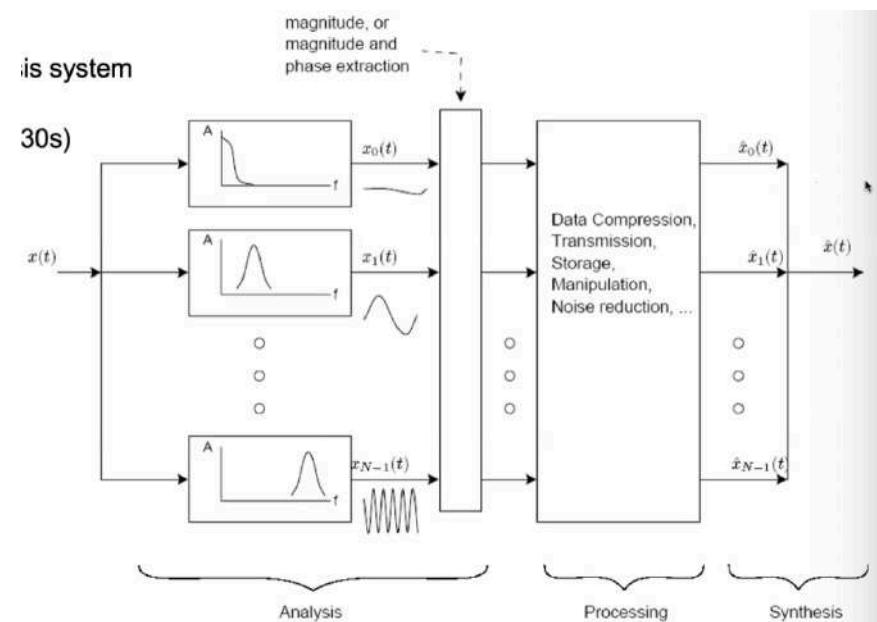
It is basically the concept behind the vocoder idea.

Let us analyse it with an example:

A parallel line of pass band filters selects one frequency portion of the original signal. The signals are sent into a block that extract magnitude or magnitude and phase. This block defines the type of vocoder we are using:

- If it determines only the magnitude of the signal in each filter banks it is called **channel vocoder**
- If it determines both magnitude and phase it is called **phase vocoder**

Analysis $\Rightarrow$ Synthesis $\downarrow$	STFT	Filterbank
I-STFT	Easy	Hard
Bank of Oscillators	Hard	Easy



Then we can process the signals by compressing it, manipulating, reducing the noise and making the data suitable to be transmitted or stored.

If we want to synthetise the signal, we just have to sum all the contributions together.

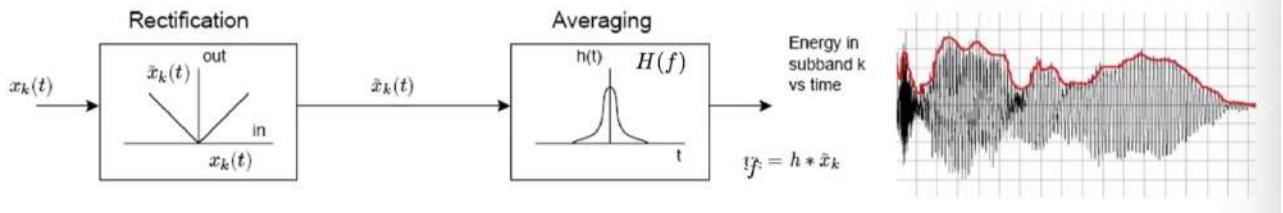
**Resolution condition:** in order to make everything work, the assumption that need to be done is that for each filter of a bank of filter there is only one sinusoidal component: we have no more than one sinusoid (with time-varying parameters) per channel. This allows us to express the signal in the  $k^{th}$  sub-band as

$$x_k(t) = a_k(t) \cos[\omega_k t + \phi_k(t)]$$

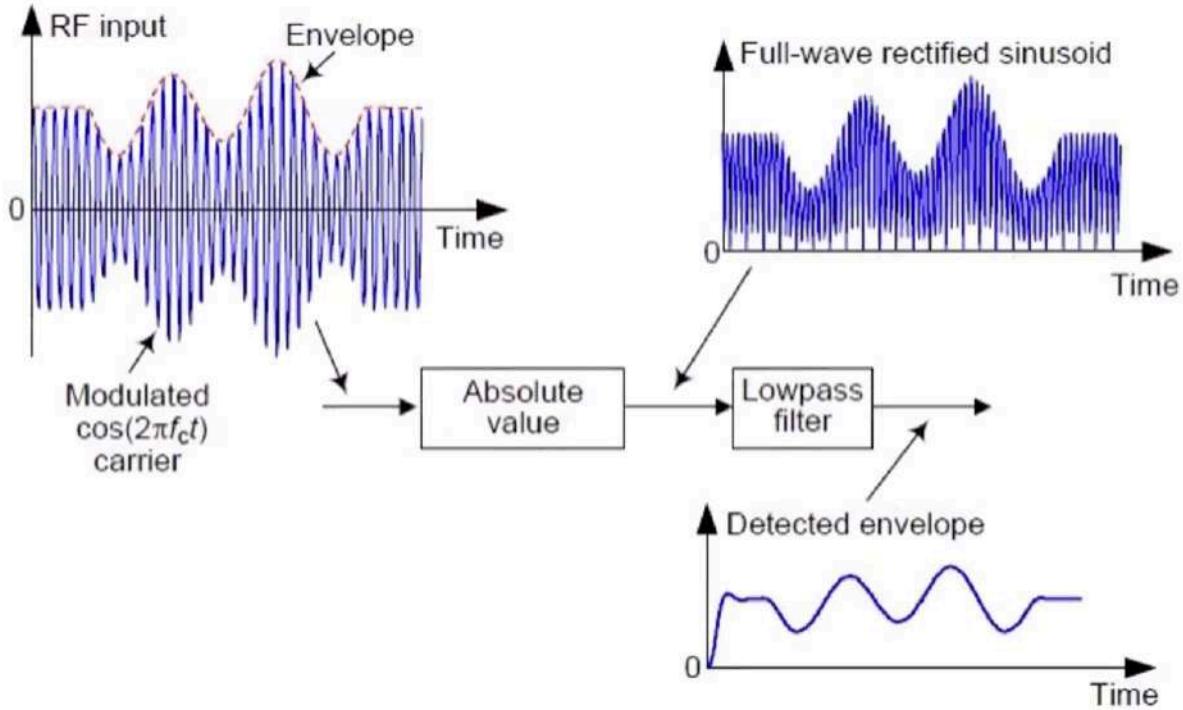
Where:

- $a_k(t)$  is an amplitude modulation term;
- $\omega_k$  is a fixed channel centre frequency;
- $\phi_k(t)$  is a phase modulation term;

As long as the amplitude is concern (channel vocoder), the things are very easy. We simply need to apply an **envelope follower** to each sub-band. Specifically, this can be done by rectification and subsequent low-pass filtering. This produces an approximation of energy in each sub-band. This approach has been applied in parametric speech models



In this example we show how to obtain the detected envelope by rectifying the signal.



The low pass filter is usually obtained with a RC circuit.

Now we have to analyse the “lazy choice”: we know that our filter is centred at a frequency  $\omega_k$ . From this derives that ideally the sinusoids at the output of the filter must have frequency at  $\omega_k$ . So, we can forget to synthetise everything else with the bank of oscillators! To synthetise the  $k$ -th sinusoid the pieces of information we will need will be the frequency  $\omega_k$  and a term that represents how the amplitude of this sinusoid changes in time  $a_k(t)$ .

But in reality, the filter is not thin and narrow, this means that it will return not only one single sinusoid at a frequency  $\omega_k$  but also some others: in order to properly resynthesise the actual sinusoid, we will need also a frequency deviation term  $\Delta\omega_k(t)$  that represents how much the frequency selected by the filter is far from  $\omega_k$  at a certain instant. This deviation here can be interpreted as the variation of the phase  $\phi_k(t)$ .

$$\Delta\omega_k(t) \triangleq \frac{d}{dt} \phi_k(t)$$

Because the frequency is the angular coefficient of the phase.

Let us dwell on this concept with an example:

we supposed to have a certain sinusoid that oscillates at a constant frequency  $\omega_k$ .

$$y(t) = A \cos(\omega_k t)$$

The phase of this sinusoid is  $\phi_k(t) = 0$ .

We now change the frequency with a term that is function of time: this means that  $\omega_k \rightarrow \omega_k + \Delta\omega_k$  and that the oscillation becomes faster or slower than  $\omega_k$  depending on the contribute of  $\Delta\omega_k$  at that instant.

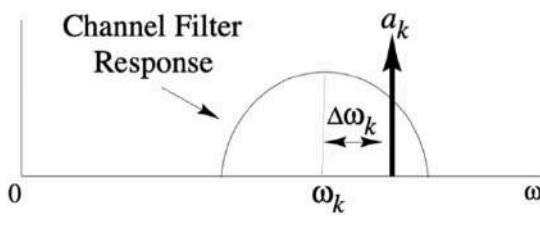
$$y_1(t) = A \cos((\omega_k + \Delta\omega_k(t))t)$$

Now the phase is  $\phi_k(t) = (\omega_k + \Delta\omega_k(t))t$  that, derived becomes

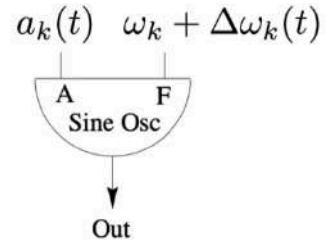
$$\frac{d}{dt}\phi_k(t) = \frac{d}{dt}[(\omega_k + \Delta\omega_k(t))t] = \omega_k + \Delta\omega_k(t)$$

Non capisco perché vada via  $\omega_k$

### Analysis Model



### Synthesis Model



So, we need some additional information on the phase. How can we retrieve it?

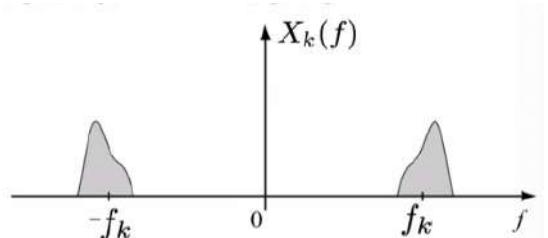
Let us consider the output signal  $x_k(t)$  from the filter (so one sinusoid with others spectral components). For what we said, this signal will be equal to

$$x_k(t) = a_k(t) \cos(\omega_k t + \phi_k(t))$$

That decomposed by means of the Euler formula becomes

$$x_k(t) = \frac{a_k(t)}{2} \{ e^{j(\omega_k t + \phi_k(t))} + e^{-j(\omega_k t + \phi_k(t))} \}$$

Whose module can be represented on a graph by two frequential components centred in  $\pm f_k = \pm \frac{\omega_k}{2\pi}$ .

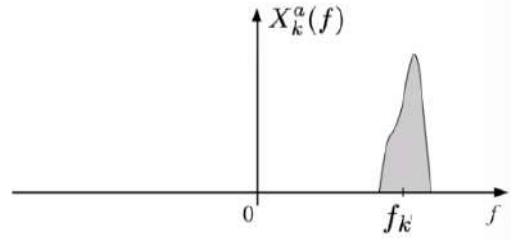


We consider now a new signal called **analytic signal**.

According to the translation property, we know that if we multiply a certain signal by a complex exponential, we shift all the spectral components of the signal of a quantity expressed by the complex exponential's frequency. It can be easily demonstrated with the duality principle starting from  $x(t - t_0)$  and noticing that it returns a multiplication between the transform of the signal and a complex exponential.

So, if we take the spectrum  $a_k$  and we multiply it for the complex exponential  $e^{j(\omega_k t + \phi_k(t))}$  we get the following result.

$$x_k^a(t) = a_k(t)e^{j(\omega_k t + \phi_k(t))}$$



Now, we shift it back to the origin obtaining the so called **base-band equivalent signal** (the same analytic signal but shifted in base-band)

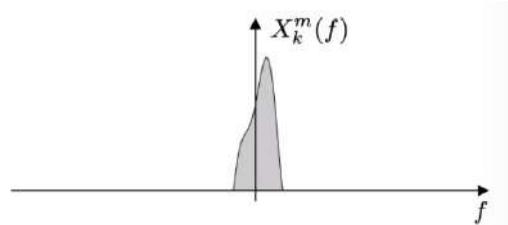
$$x_k^m(t) = e^{-j\omega_k t} x_k^a(t) = a_k(t)e^{j\phi_k(t)}$$

This result can be analysed from two different viewpoints:

1. We split the exponential sum of  $\omega_k t + \phi$  into the product of two exponentials and the frequency term goes away

$$x_k^m(t) = e^{-j\omega_k t} x_k^a(t) = a_k(t)e^{j\omega_k t} e^{j\phi_k(t)} e^{-j\omega_k t} = a_k(t)e^{j\omega_k t - j\omega_k t} e^{j\phi_k(t)} = a_k(t)e^{j\phi_k(t)}$$

2. We shift everything to the left by a quantity that makes irrelevant the constant shifting in frequency.



If we do this, we obtain a signal whose is exactly equal to

$\phi_k(t)$  that is exactly the phase that we want to estimate to find the changing in frequency.

Estimating the phase from a signal like this one it is easier rather than estimate it from the signal from which we started. We basically have to take the real part and the imaginary part of the function (quale?) and we compute the inverse tangent of the ratio of this results.

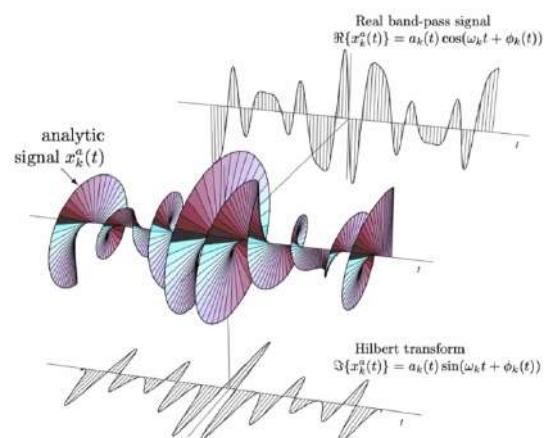
But how can we find the analytic function?

We are lucky since there exists a Transform, being the Hilbert transform, that given your cosine allows us to estimate which is the imaginary part to be added to this cosine in order to get the complex exponential we miss. Why do we need just the imaginary part? Because evaluating the real part of the analytic signal we get exactly the original signal (that we know).

$$\begin{aligned} \Re\{x_k^a(t)\} &= \Re\{a_k(t)e^{j(\omega_k t + \phi_k(t))}\} \\ &= \Re\{a_k(t) \cos(\omega_k t + \phi_k(t)) \\ &\quad + j \sin(\omega_k t + \phi_k(t))\} = \\ &= a_k(t) \cos(\omega_k t + \phi_k(t)) \end{aligned}$$

As we said, we need to find which is the imaginary part to be added to the signal we know to obtain in order to find the analytic function.

This is exactly what the Hilbert transform does: given a real signal, it returns the imaginary part that we have to add to this signal in order to find a particular function called



Analytic function that shifted to the origin allows us to evaluate the phase of the original signal.

The transform is defined as the response of the real  $-\frac{\pi}{2}$  phase-shifter (a.k.a. Hilbert Filter), whose frequency response is

$$G_H(f) = -j \operatorname{sgn}(f) = e^{-j\frac{\pi}{2}\operatorname{sgn}(f)}$$

Whose corresponding impulse response is  $g_H(t) = \frac{1}{\pi t}$ .

So, given a general signal  $x(t)$  we can express in closed form its Hilbert transform by:

$$\hat{x}(t) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{x(\tau)}{t - \tau} d\tau$$

Let us see why the transform is equal to the imaginary part of the analytic signal.

We want to demonstrate that

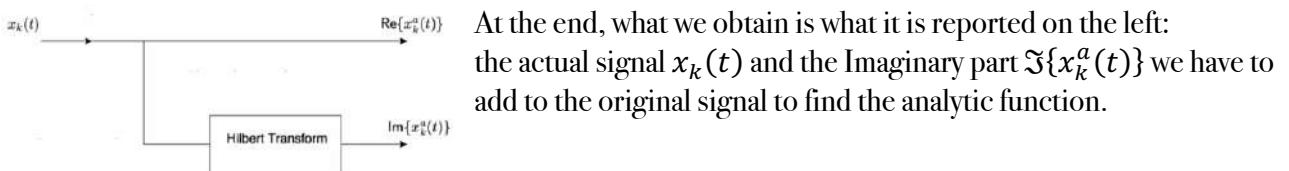
$$x_a(t) = x(t) + j\hat{x}(t)$$

In order to do so, let us compute the Fourier Transform of the imaginary part of the analytic signal by considering its frequency decomposition

$$\begin{aligned} \mathcal{F}\{\Im\{x_a(t)\}\} &= [\dots] = \frac{1}{2j} [X_a(f) - X_a^*(f)] = \frac{1}{2j} [1(f) - 1(-f)] X(f) = -j \operatorname{sgn}(f) X(f) = \\ &= G_H(f) X(f) \end{aligned}$$

Since the Hilbert filter is an all pass filter, we can recover the inverse Hilbert filter by simply finding the reciprocal of its frequency response. So,

$$\frac{1}{G_H(f)} = j \operatorname{sgn}(f) \xrightarrow{\mathcal{F}^{-1}} -\frac{1}{\pi t}$$



So let us summarise and conclude the whole process:

1. We start from a cosine of which we want to estimate the phase

$$x_k(t) = a_k(t) \cos(\omega_k t + \phi_k(t))$$

2. We compute the Hilbert Transform of this cosine finding the imaginary part that we have to add to it in order to find the analytic signal  $x_k^a(t) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{a_k(\tau) \cos(\omega_k \tau + \phi_k(\tau))}{t - \tau} d\tau$

3. We shift the analytic signal towards base-band obtaining the base-band equivalent signal.

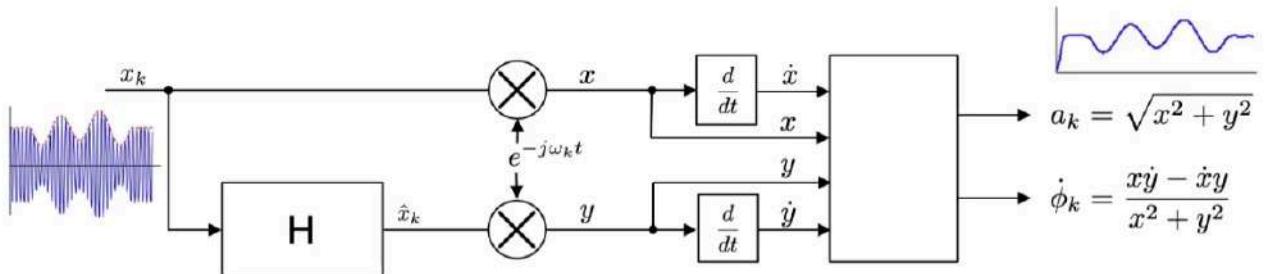
4. We estimate the phase as follows

$$\begin{aligned}\phi_k(t) &= \angle x_k^m(t) = \angle \{x_k^a(t)e^{-j\omega_k t}\} = \angle x_k^a(t) + \angle e^{-j\omega_k t} = \angle x_k^a(t) - j\omega_k t = \\ &= \tan^{-1} \left( \frac{\Im\{x_k^a(t)\}}{\Re\{x_k^a(t)\}} \right) - \omega_k t\end{aligned}$$

Having to use the inverse tangent, however, is not our best option due to the burden of having to compute a transcendental function and to solve some ambiguities associated to the inversion of tangent. We will see that it is more convenient to extract the instantaneous frequency deviation instead of the phase. As we have seen, we can compute it as the derivative of the phase. We remind that we are allowed to proceed in this way as we are working with baseband signals.

$$\begin{aligned}\Delta\omega_k(t) &\triangleq \frac{d}{dt} \angle x_k^m(t) = \dot{\phi}_k(t) = \frac{d}{dt} \arctg \left( \frac{\Im\{x_k^m(t)\}}{\Re\{x_k^m(t)\}} \right) = \\ &= \frac{\frac{d}{dt} \left( \frac{\Im\{x_k^m(t)\}}{\Re\{x_k^m(t)\}} \right)}{1 + \left( \frac{\Im\{x_k^m(t)\}}{\Re\{x_k^m(t)\}} \right)^2} = \frac{\Re\{x_k^m(t)\}^2 \left[ \frac{d}{dt} \frac{\Im\{x_k^m(t)\}}{\Re\{x_k^m(t)\}} - \frac{\Im\{x_k^m(t)\}}{\Re\{x_k^m(t)\}} \frac{d}{dt} \Re\{x_k^m(t)\} \right]}{\Re\{x_k^m(t)\}^2 + \Im\{x_k^m(t)\}^2} = \\ &= \frac{\Re\{x_k^m(t)\} \cdot \frac{d}{dt} \Im\{x_k^m(t)\} - \frac{d}{dt} \Re\{x_k^m(t)\} \cdot \Im\{x_k^m(t)\}}{\Re\{x_k^m(t)\}^2 + \Im\{x_k^m(t)\}^2}\end{aligned}$$

So, at the end of the day, this is the scheme we have to follow:



Inherent issues:

- We cannot have more than one sinusoid per sub-band (can lead to lots of filters)
- Poor model for signal with transient or sharp attacks
- Inconvenient for non-harmonic signals (lines not uniformly spaced in frequency)
- Inefficient model for signal with noise-like qualities
- Not an identity system (unless phase is retained, and no data reduction is done)
- Computationally expensive

Before going on, let us wrap up everything very quickly:

In terms of analysis, we have STFT, DTFT and also the one based on filterbanks.

In terms of synthesis, we can reverse the STFT or use oscillator banks.

So, let us dig more into the differences between these methods:

our aim is to capture the temporal evolution of the signal's spectrum. In order to do so, we need a tool that computes the “local” Fourier transform at any time.

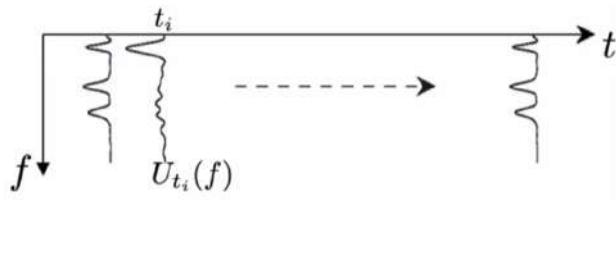
$$\begin{array}{ccc} & \text{Local Fourier} \\ & \text{Transform} \\ u(t) & \xrightarrow{\quad} & U(t, f) \end{array}$$

We saw two analysis tools that do that:

**STFT**

$$u(t) \xrightarrow{\text{STFT}} U_t(f)$$

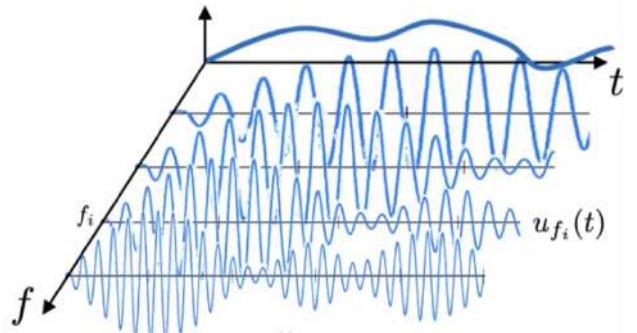
*Privileging frequency over time*



**Filterbanks**

$$u(t) \xrightarrow{\text{FB}} u_f(f)$$

*Privileging time over frequency*



With the STFT we look very deeply what happens in frequency domain while with the Filterbank method we privilege what happens in time domain. The result is almost the same.

If we do the math, we notice that they corresponds.

Let us consider the STFT of a certain signal:

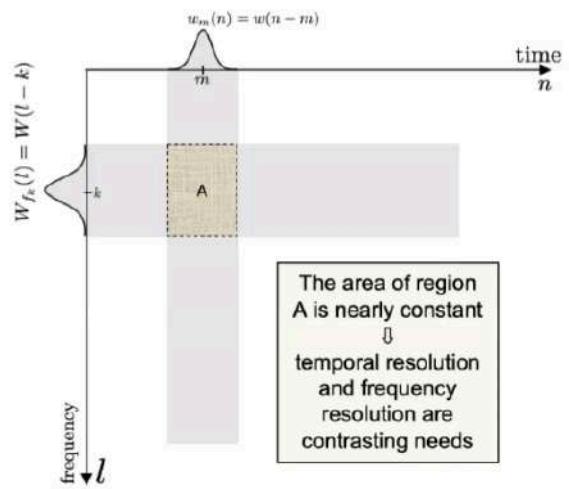
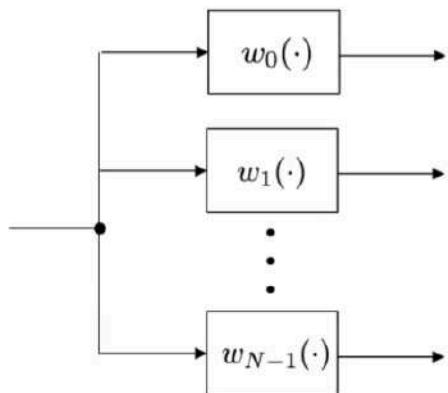
$$x(nT) \xrightarrow{\text{STFT}} X(t_i, f_k) = \sum_n w_a(n) x(t_i + nT) e^{-j2\pi f_k n T}$$

Let us consider the DTFT of the windowed signal (window starting from sample m)

$$\begin{aligned} X_m(f_k) &= X(m, f_k) = \sum_{n=-\infty}^{\infty} w(n) x(m+n) e^{-j2\pi f_k n N} \stackrel{r \triangleq m+n}{=} \\ &= \sum_{r=-\infty}^{\infty} w(r-m) x(r) e^{-j2\pi f_k (r-m) N} \stackrel{\substack{\text{If the window is even} \\ \text{it can be flipped left} \\ \text{to right}}}{} \\ &\quad w(\alpha)=w(-\alpha) \end{aligned}$$

$$= \sum_{r=-\infty}^{\infty} w(m-r) x(r) e^{-j2\pi f_k (r-m) N} \stackrel{w_{f_k}(n)=w(n)e^{-j2\pi f_k n N}}{=} \sum_{r=-\infty}^{\infty} w_{f_k}(m-r) x(r) = w_{f_k} * x(m)$$

We notice that applying a STFT is like filtering the signal with a filter shifting filter (in fact  $\mathcal{F}\{w(n)e^{-j2\pi f_k n N}\} = W(f - f_k)$ ).



## Signal modelling on Matlab

In today's lecture we will see something about modelling basic signals.

We start by considering a sinusoid:

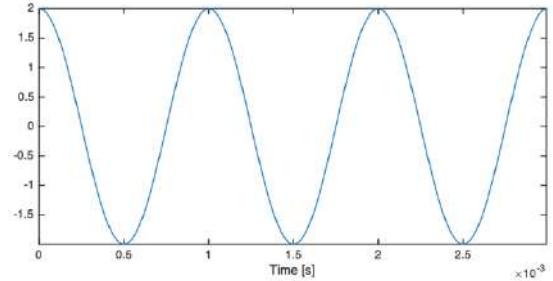
$$x(t) = A \sin(2\pi ft + \phi)$$

```

Fs = 44100; % sample frequency
Ts = 1/Fs; % sample period
t = 0 : Ts : 0.003; % time axis
A = 2; % amplitude of the sinusoid
f = 1000; %H % frequency of the sinusoid
phi = pi/2; %rad % phase of the sinusoid
x = A*sin(2*pi*f*t + phi); % compute the signal samples
plot(t,x); axis tight % plot the signal
xlabel('Time [s]');

```

We start reminding that MATLAB stands for mathematical laboratory: so we can imagine that this program is perfect when we want to work with matrixes. That's why we always have to deal with arrays.



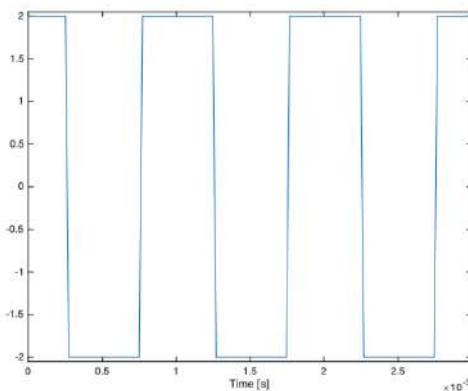
Other interesting signals are :

The square wave

```

x = A*square(2*pi*f*t + phi);
% compute the signal samples

```

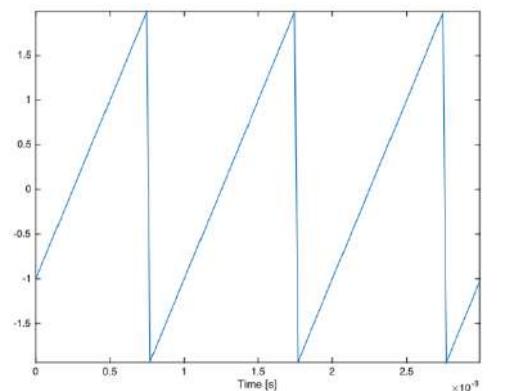


The Sawtooth wave

```

x = A*sawtooth(2*pi*f*t + phi);
% compute the signal samples

```



This second one is useful when we want to analyse the transient of a system.

### Basical operations with audio:

- Opening an audio file

```
[s, fs] = wavread('myAudioFile.wav');
[s, fs] = audioread('myAudioFile.wav'); % newest
Matlab versions
```

- Writing an audio file

```
x = randn(10000,1);
x = 0.99*x/max(abs(x)); % normalization in the
range (-1,1)
% Write at 44100Hz, 16 bits-per-sample
wavwrite(x,44100,16,'noise.wav');
audiowrite('noise.wav',x,44100,'BitsPerSample',16);
% newest versions
```

- Reproducing an audio file

```
wavplay(x,fs); % only M.S. Windows
sound(x,fs); % cross-platform
soundsc(x,fs); % normalizes the signal before
playing
```

With sound sc the signal is  
normalised so the amplitude is  
maximised

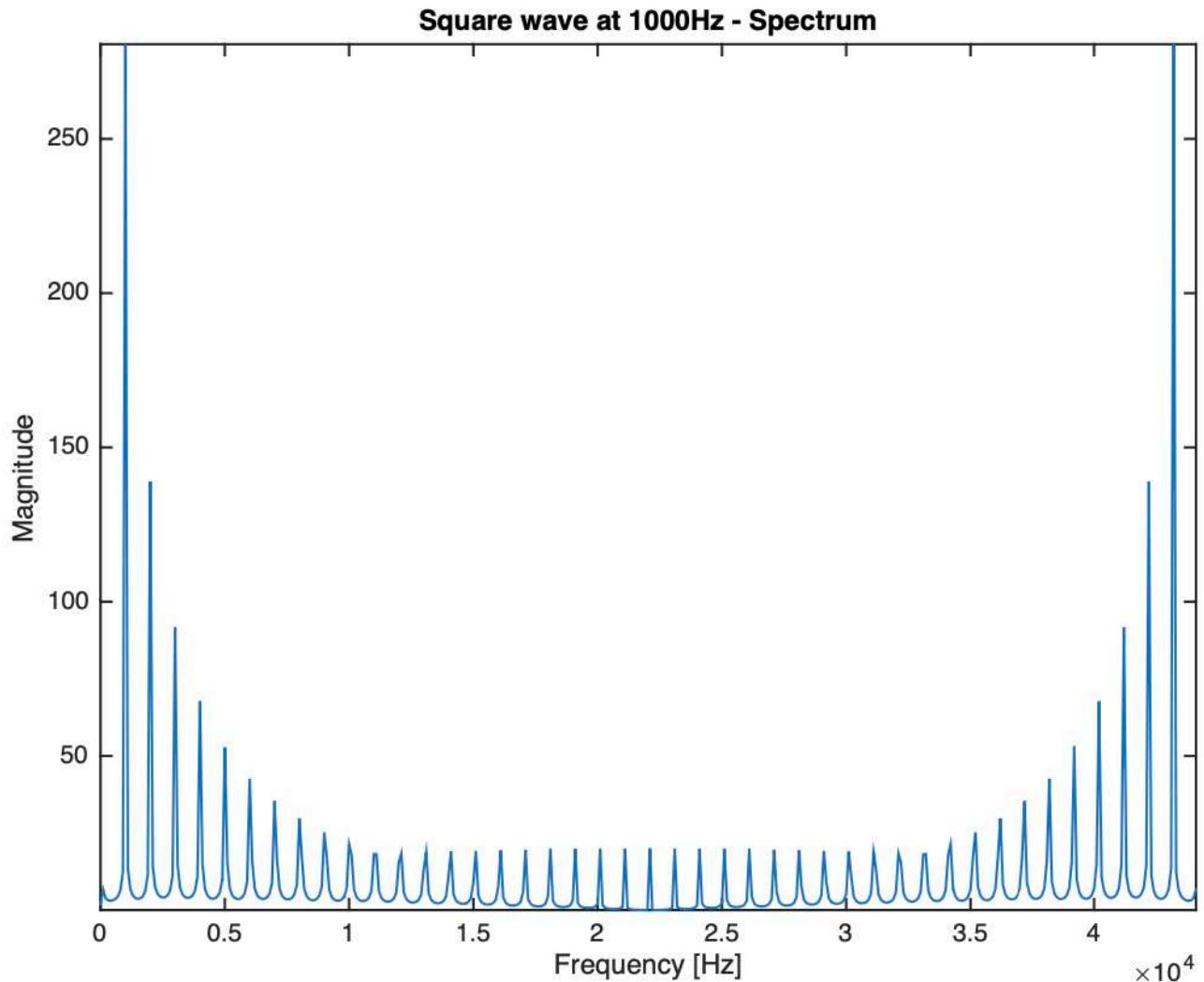
### Analysis of the signal:

The DFT is defined as a matrix operation in which the signal is projected on a series of sinusoidal bases.  
We evaluate it by means of the fft.

```
X = fft(x);
```

A standard way to represent signals is to use capital letters for frequency domain and small one for time domain.

```
X = fft(x); % compute the FFT
L = length(X); % FFT length
f_ax = 0 : Fs/(L-1) : Fs; % frequency axis
plot(f_ax,abs(X)); axis tight
xlabel('Frequency [Hz]');
ylabel('Magnitude');
title('Square wave at 1000Hz - Spectrum')
```



Example of the DFT of a squared wave, we can notice how the peaks are in correspondence of odd harmonics. We can notice also that the spectre is symmetric (it is represented from 0 to two times the fundamental).

The FFT algorithm is really fast only when  $L$  is a power of 2, i.e. when  $L = 2^M \rightarrow$  complexity in the order of  $O(L \log L)$

If  $L \neq 2^M \rightarrow$  Complexity is in the order of  $O(L^2)$ . To exploit the fastness of the fft we must zero pad. By zero padding, we reach a total length of  $N = L + Z = 2^M$

```
N = 512; % power of 2 next to L=442
Z = N - L; % number of zeros to be added
x_ = [x zeros(1,Z)]; % new signal of length Z + L
X_ = fft(x_);
```

We concatenate the singal x with a series of Z zeroes

Alternatively, we can embed the zero padding in the fft

```
N = 512; % power of 2 next to L=442
X_ = fft(x,N); % zero padding embedded in the fft function
```

Other functions that can become useful in terms of signal analysis are

```

X = fft(x);
X_abs = abs(X);           % magnitude
X_phi = phase(X);        % phase
y = ifft(X);             % inverse Fast Fourier Transform

```

## Modelling the noise

It is often useful to model the generation of noise:

- To simulate a noisy process;
  - To measure the acoustic response of an environment;

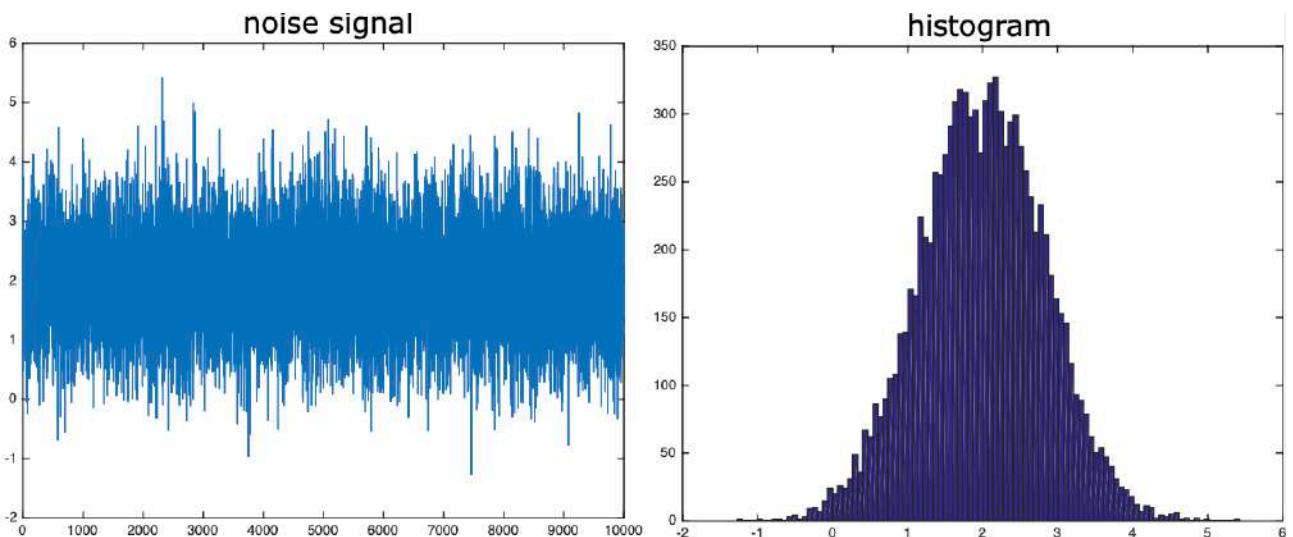
The most common noise type is white noise; it can be simulated through random data extracted from a Gaussian distribution as it is done in the following example:

```

mu =2; %mean sig2 = 0.7; % variance

% generate random samples from Gaussian distribution
x = sqrt(sig2)*randn(10000,1) + mu;
figure(1)
plot(x)      % plot the noise
figure(2)
hist(x,100) % histogram (estimate the pdf)

```



We notice that the histogram of this realisation follows a Gaussian distribution centred around 2, as we expected.

Example: 1000 Hz sinusoid with additive zero mean white noise, SNR = 10dB

```

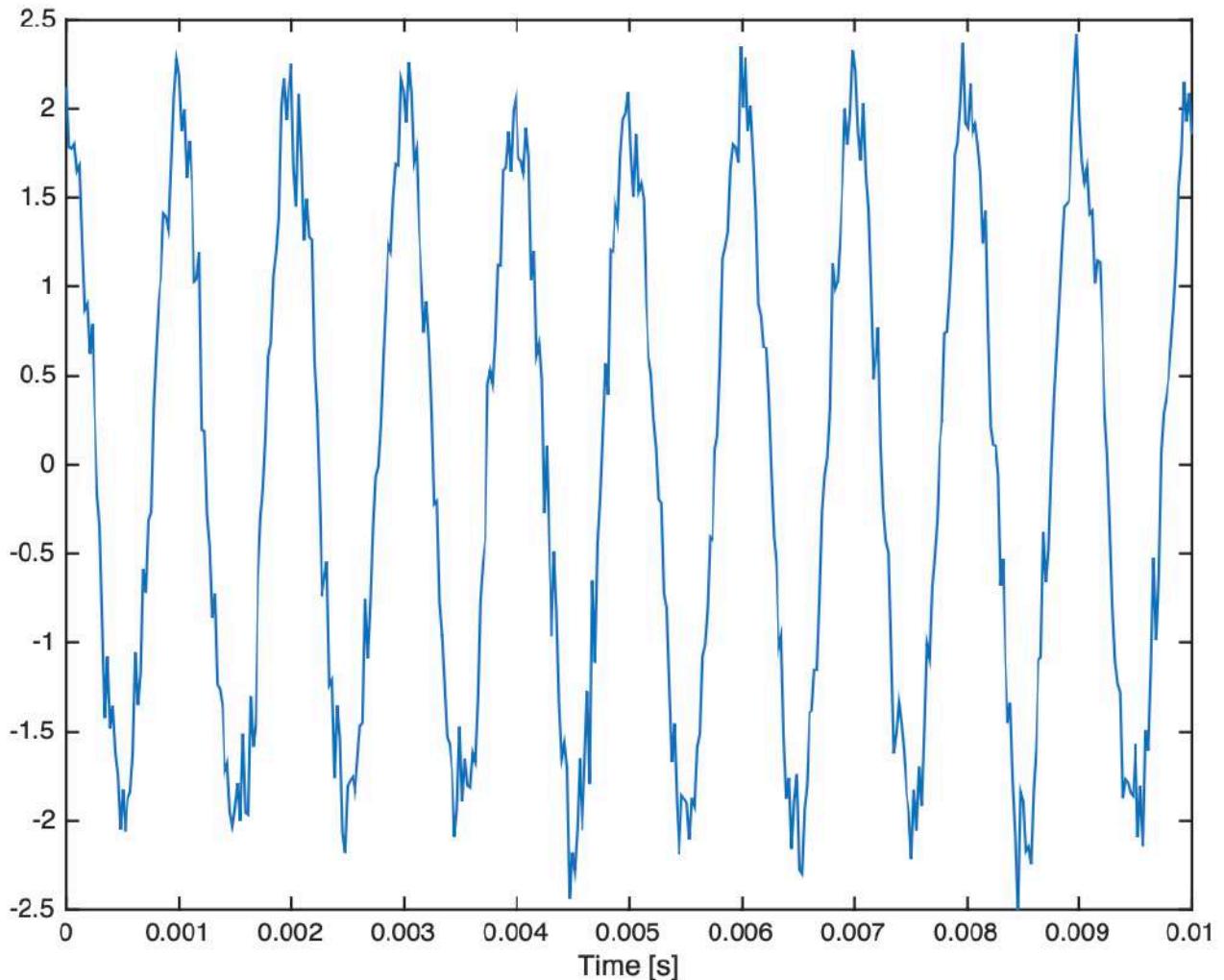
Fs = 44100; % sample frequency [Hz]
Ts = 1/Fs; % sample period [s]
t = 0 : Ts : 0.01; % time axis
A = 2; % amplitude of the sinusoid
f = 1000; % frequency of the sinusoid [Hz]

```

```

phi = pi/2; % phase of the sinusoid [rad]
x = A*sin(2*pi*f*t + phi); % compute the signal samples
SNR_dB = 10; % SNR
std_v = A / (10^(SNR_dB/20)); % std. dev. of noise
v = std_v*randn(size(x)); % white noise samples
s = x + v; % noisy signal
plot(t,s); axis([0 0.01 -2.5 2.5]) % plot the signal
xlabel('Time [s]');

```



## Convolution

For discrete convolution in the time domain we have

$$y(n) = \sum_j x(j)h(n-j+1)$$

And it is computed by the line

```
y = conv(x, h); % convolution of x and h
```

If we have to specify the coefficients of a finite length filter like the following

$$y(n) = b(0)x(n) + b(1)x(n-1) + \dots + b(N_b)x(n-N_b)$$

We write

```
y = filter(b, 1, x); % filtering x with coefficients in
vector 'b'
```

While for an IIR filter

$$y(n) = b(0)x(n) + b(1)x(n-1) + \dots + b(N_b)x(n-N_b) - a(1)y(n-1) - \dots - a(N_a)y(n-N_a)$$

```
y = filter(b, a, x); % filt. x with coefficients in vectors
'b' and 'a'
```

## Statistics

There is also the possibility to compute statistics:

- Cross-correlation of two signals:

$$r_{xy}(n) = \sum_j x(j)y(j+n)$$

```
r_xy = xcorr(x, y); % cross-correlation of x and y
```

- Autocorrelation function:

```
r_x = xcorr(x); % auto-correlation of x
```

- Mean, variance, etc...:

```
m = mean(x); % mean of the vector x
v = var(x); % variance of the vector x
s = std(x); % standard deviation of the vector x
C = cov(X); % covariance matrix of the (NxM)
matrix X
% C is a (MxM) matrix
```

## Exercise 1

Create a figure with 4 subplots stacked vertically. They contain sinusoids with same frequency but different phases ( $0, \pi/2, \pi, 3\pi/2$ )

```
%% Figures
% Create a figure with 4 subplots stacked vertically
% They contain sinusoids with same frequency but different phases
% (0, pi/2, pi, 3pi/2)

figure(); % it opens a pop-up where the signal will be plotted
N = 4;

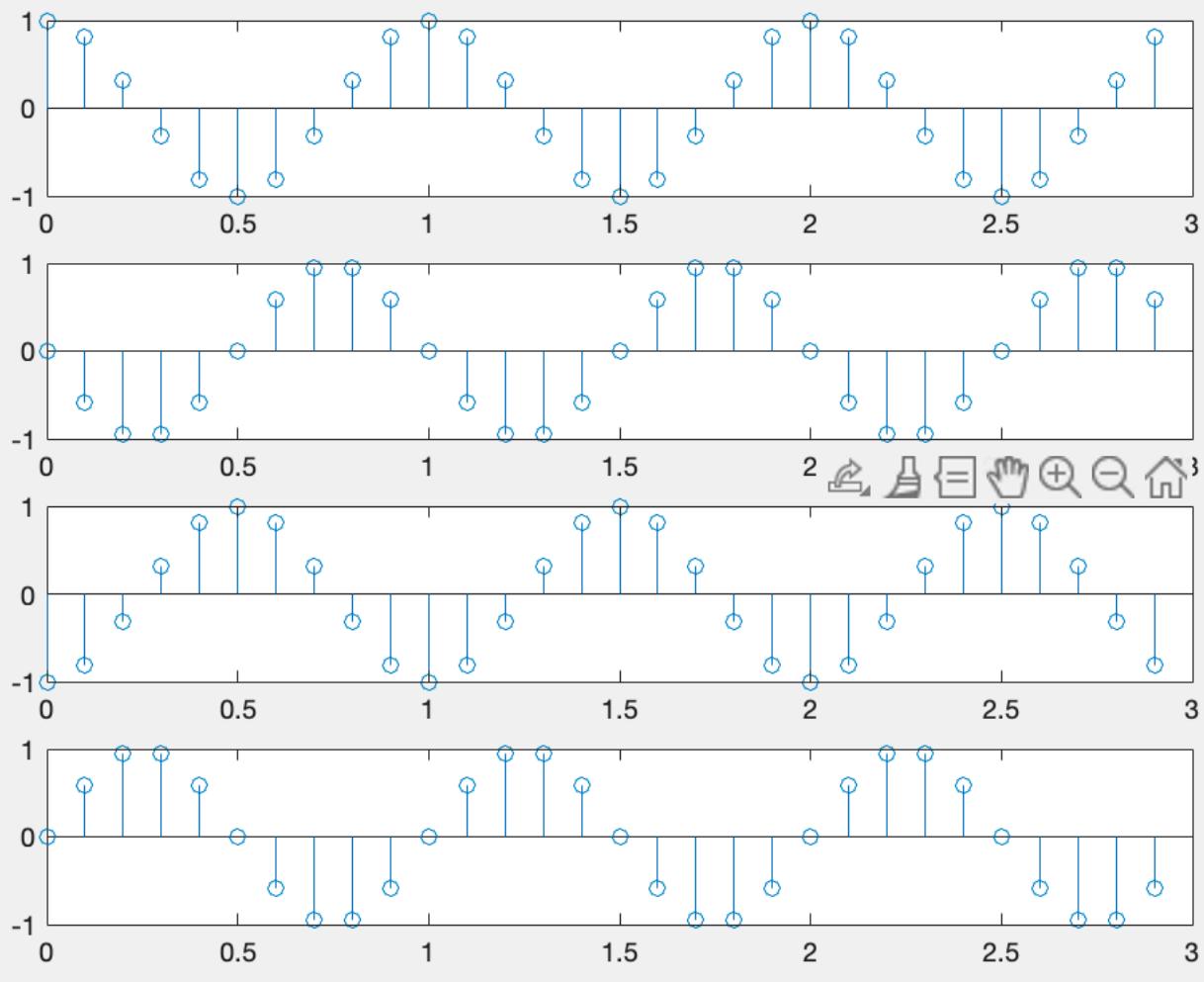
f = 1; %Hz
Fs = 10;
duration = 3; %s

Ts = 1/Fs;
t = 0:Ts:(duration-Ts); % defines the time axis: it goes from 0 to duration - 1
with a step equal to the sampling period
```

```

for ii = 1:N
    subplot(N, 1, ii);
    phi = (ii - 1)/N * 2 * pi; %we exploit the index in a smart way to define
    %the three phases
    x = cos(t*2*pi*f + phi);
    stem(t, x); %stem is used to represent discrete sequences
end

```



### Exercise 2

Simulate the impact of using a wrong sampling frequency.

Write a script that:

- Simulate 4 "continuous" cosine signals
- Sampling period = 0.001s
- Duration = 5 s
- Frequencies:
  - $f_1 = 0.2 \text{ Hz}$
  - $f_2 = 0.2 + 1/\Delta T \text{ Hz}$
  - $f_3 = 0.2 + 2/\Delta T \text{ Hz}$
  - $f_4 = 0.2 + 3/\Delta T \text{ Hz}$
- $\Delta T = 0.5 \text{ s}$

Sample the "continuous" signals with a sampling period = 0.5 s and plot it. Make a figure with 4 sub-plots containing the "continuous" and the relative sampled signals

```

%%%%%
% Introduction to Matlab for audio processing
% DAAP course
% 2024
% Mirco Pezzoli
% Credits to Federico Borra
% Exercise 2
% %%%%%%%

clear all
clearvars
clc
%% Aliasing
% Write a script that
%   - Simulate 4 "continuous" cosine signals
%   - Sampling period = 0.001s
%   - Duration = 5 s
%   - Frequencies:
%     f1=0.2Hz
%     f2 =0.2+1/DT Hz
%     f3 =0.2+2/DT Hz
%     f4 =0.2+3/DT Hz
%   - DT=0.5s
% Sample the "continuous" signals
%   - Sampling period = 0.5 s
% Plot
%   Make a figure with 4 sub-plots containing the ?continuous? and the
%   relative sampled signals
%% Parameters
Ts = 0.001; %s
duration = 5; %s
dt = 0.001; %s
Fs_c = 1/dt; %Hz
t = 0:dt:(duration-dt);

DT = 0.5;
T = 0:DT:(duration-DT); %time axis for the low sampling frequency

f1 = 0.2+1/DT; %Hz
f2 = 0.2+1/DT; %Hz
f3 = 0.2+2/DT; %Hz
f4 = 0.2+3/DT; %Hz

%% "Continuous" signals
x1 = cos(2 * pi * f1 * t);
x2 = cos(2 * pi * f2 * t);
x3 = cos(2 * pi * f3 * t);
x4 = cos(2 * pi * f4 * t);

%% Sampled signals
x1_n = cos(2 * pi * f1 * T);
x2_n = cos(2 * pi * f2 * T);
x3_n = cos(2 * pi * f3 * T);
x4_n = cos(2 * pi * f4 * T);

%% Plot
figure();
subplot(1,4,1);

```

```

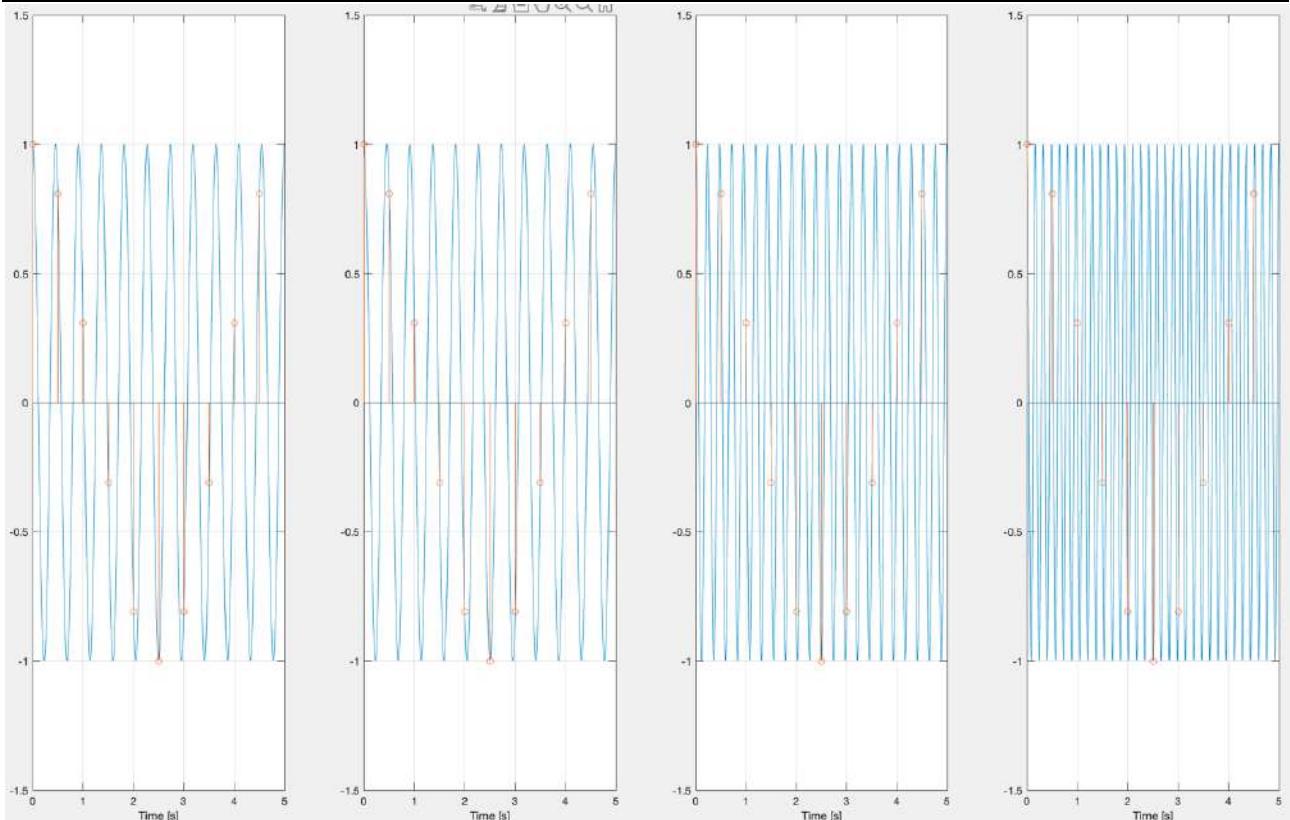
plot(t, x1), hold on; %first domain and module of the function. Hold on keeps
the figure active in order to plot again in it.
stem(T, x1_n);
xlabel('Time [s]');
grid on, axis([0, 5, -1.5, 1.5]);

subplot(1,4,2);
plot(t, x2), hold on; %first domain and module of the function. Hold on keeps
the figure active in order to plot again in it.
stem(T, x2_n);
xlabel('Time [s]');
grid on, axis([0, 5, -1.5, 1.5]);

subplot(1,4,3);
plot(t, x3), hold on; %first domain and module of the function. Hold on keeps
the figure active in order to plot again in it.
stem(T, x3_n);
xlabel('Time [s]');
grid on, axis([0, 5, -1.5, 1.5]);

subplot(1,4,4);
plot(t, x4), hold on; %first domain and module of the function. Hold on keeps
the figure active in order to plot again in it.
stem(T, x4_n);
xlabel('Time [s]');
grid on, axis([0, 5, -1.5, 1.5]);

```



### Exercise 3 - Simulate additive noise.

Let's simulate the acquisition of a microphone signal. The signal is sampled at 8kHz, and the peak signal to noise ratio is 5dB. The acquired signal is composed of two sinusoidal tone with frequency 500Hz and 1000Hz respectively. The signal length is equal to 5ms. Compute the signal samples and plot it.

```

clc
close all
clearvars

%% Simulate additive noise.
% Let's simulate the acquisition of a microphone signal. The signal is
% sampled at 8kHz, and the peak signal to noise ratio is 5dB. The acquired
% signal is composed of two sinusoidal tone with frequency 500Hz and 1000Hz
% respectively. The signal length is equal to 5ms.
% Compute the signal samples and plot it.

%% Parameters
Fs = 8000; %Hz
PSNR = 5; %dB

A = 0.5; %signal amplitude
f1 = 500; %Hz
f2 = 1000; %Hz
duration = 5e-3; %s

t = 0 : 1/Fs : (duration - 1/Fs);

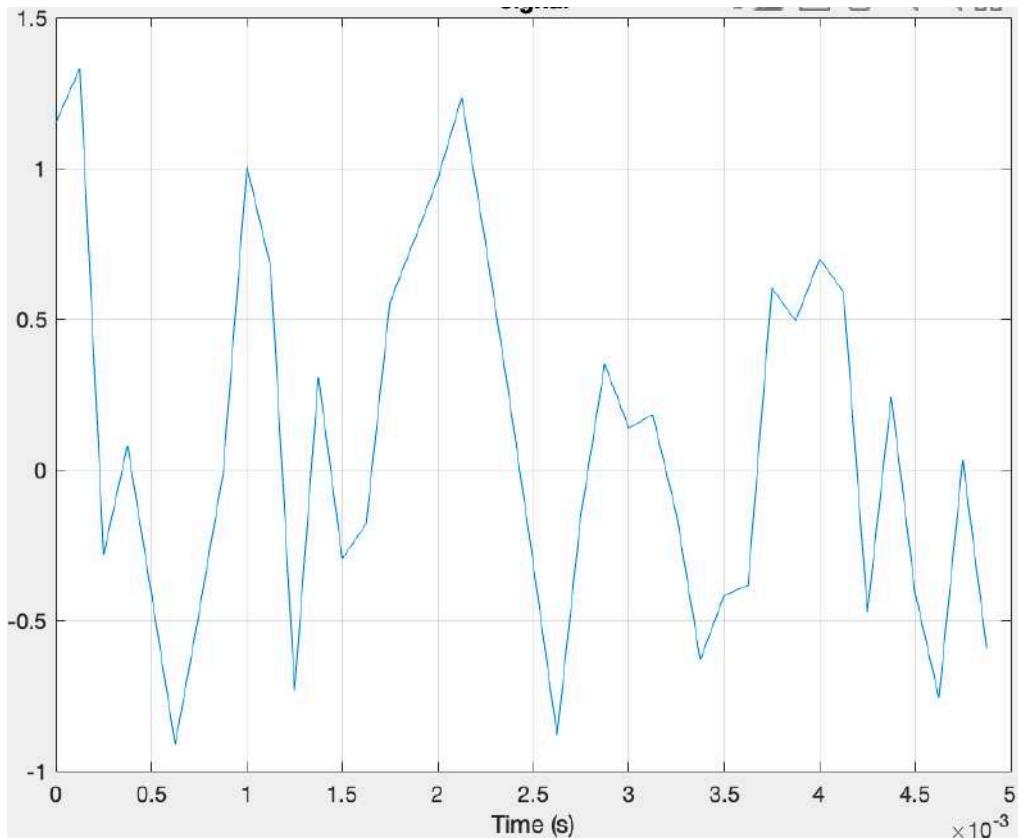
x1 = A * cos(2 * pi * f1 * t);
x2 = A * cos(2 * pi * f2 * t);
x = x1 + x2;

std_v = A / (10^(PSNR/20)); %Standard deviation

v = std_v*randn(size(x));
s = x + v;

figure();
plot(t, s);
title('Signal');
xlabel('Time (s)');
grid on;

```



## Part 2

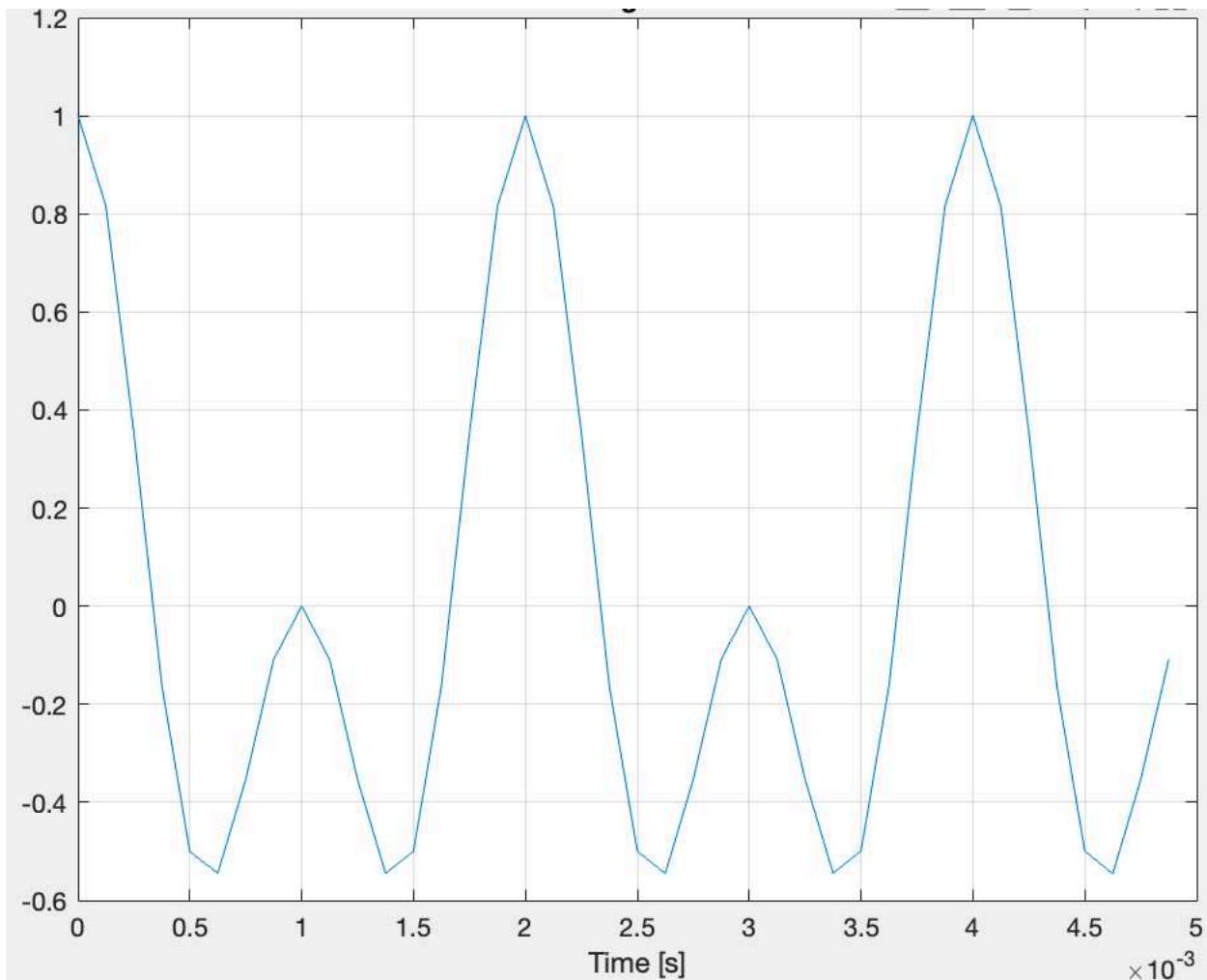
We repeat the measurement using an higher quality microphone with peak SNR of 60dB. Simulate this second acquisition and comment the result.

```
%% Part 2
% We repeat the measurement using an higher quality microphone with peak
% SNR of 60dB. Simulate this second acquisition and comment the result.

PSNR = 60;
std_v = A / (10^(PSNR/20)); %Standard deviation

v = std_v*randn(size(x));
s = x + v;

figure();
plot(t, s);
title('Signal');
xlabel('Time [s]');
grid on;
```



#### Exercise 4 - Frequency analysis

We want to inspect the frequency content of a signal. The signal is composed of a square wave at 100Hz. Choose a proper sampling frequency and compute the signal with a length of 0.05s. Plot the signal in time domain.

```

clear
close all
clc

%% Frequency analysis
% We want to inspect the frequency content of a signal. The signal is
% composed of a square wave at 100Hz.
% Choose a proper sampling frequency, and compute the signal with a
% length of 0.05s. Plot the signal in time domain.

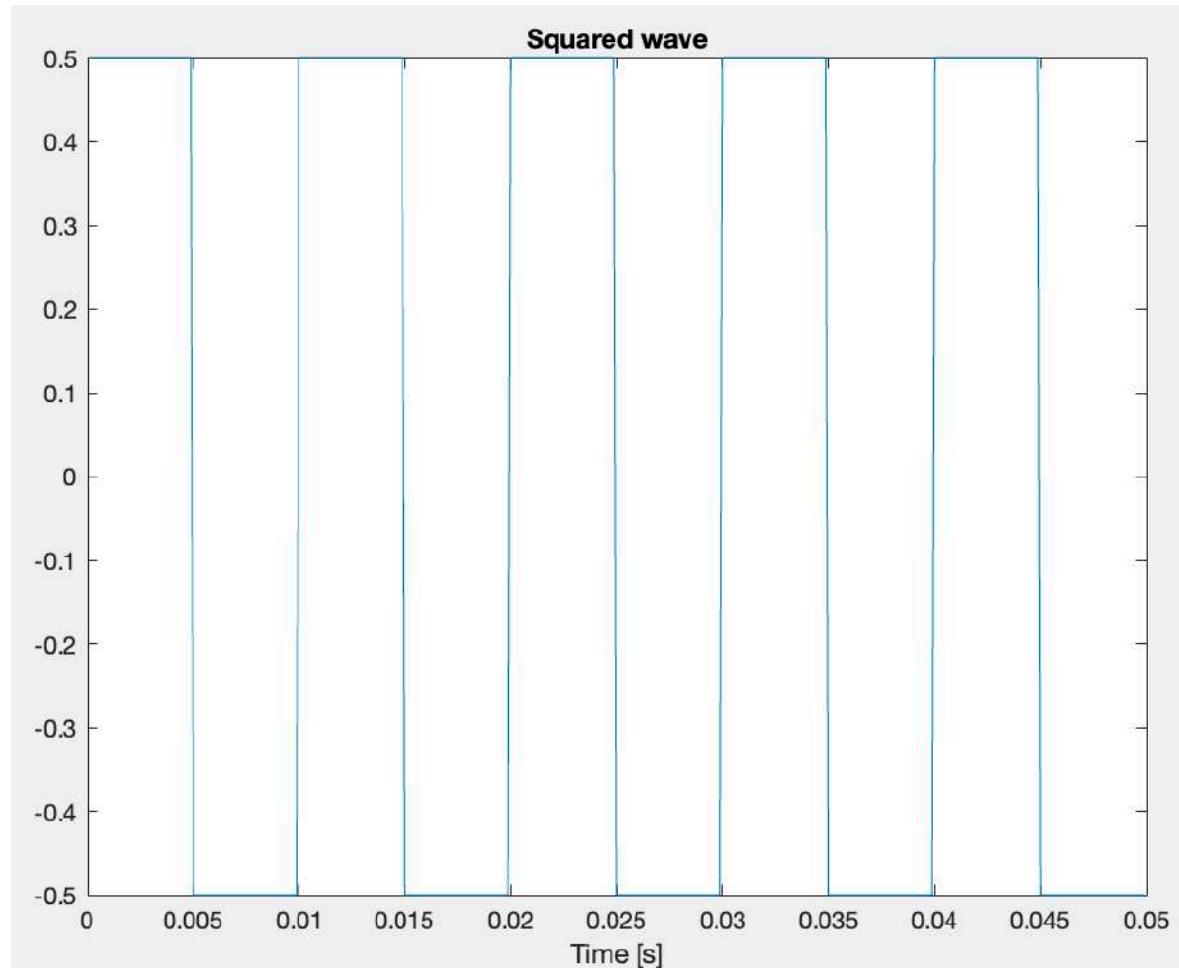
f = 100; %Hz
Fs = 4000; %Hz Nyquist Fs >= 2*fhigher
duration = 0.05; %s

```

```
t = 0 : 1/Fs : (duration - 1/Fs);
A = 0.5;

signalLen = length(t);
x = A * square(2*pi*f*t)

figure();
plot(t, x);
title('Squared wave');
xlabel('Time [s]');
```



Define in a new variable the same signal corrupted by a white noise with SNR equal to 10dB.

```
%% Define in a new variable the same signal corrupted by a white noise with
% SNR equal to 10dB.
PSNR = 10;
std_noise = A / (10^(PSNR/20)); %Standard deviation

noise = std_noise*randn(size(x)); %white noise samples
xandnoise = x + noise; %noisy signal
```

Compute the DFT of the signal

We examine the frequency spectrum of the signal, plot the magnitude and the phase of the spectrum in one figure for each signal

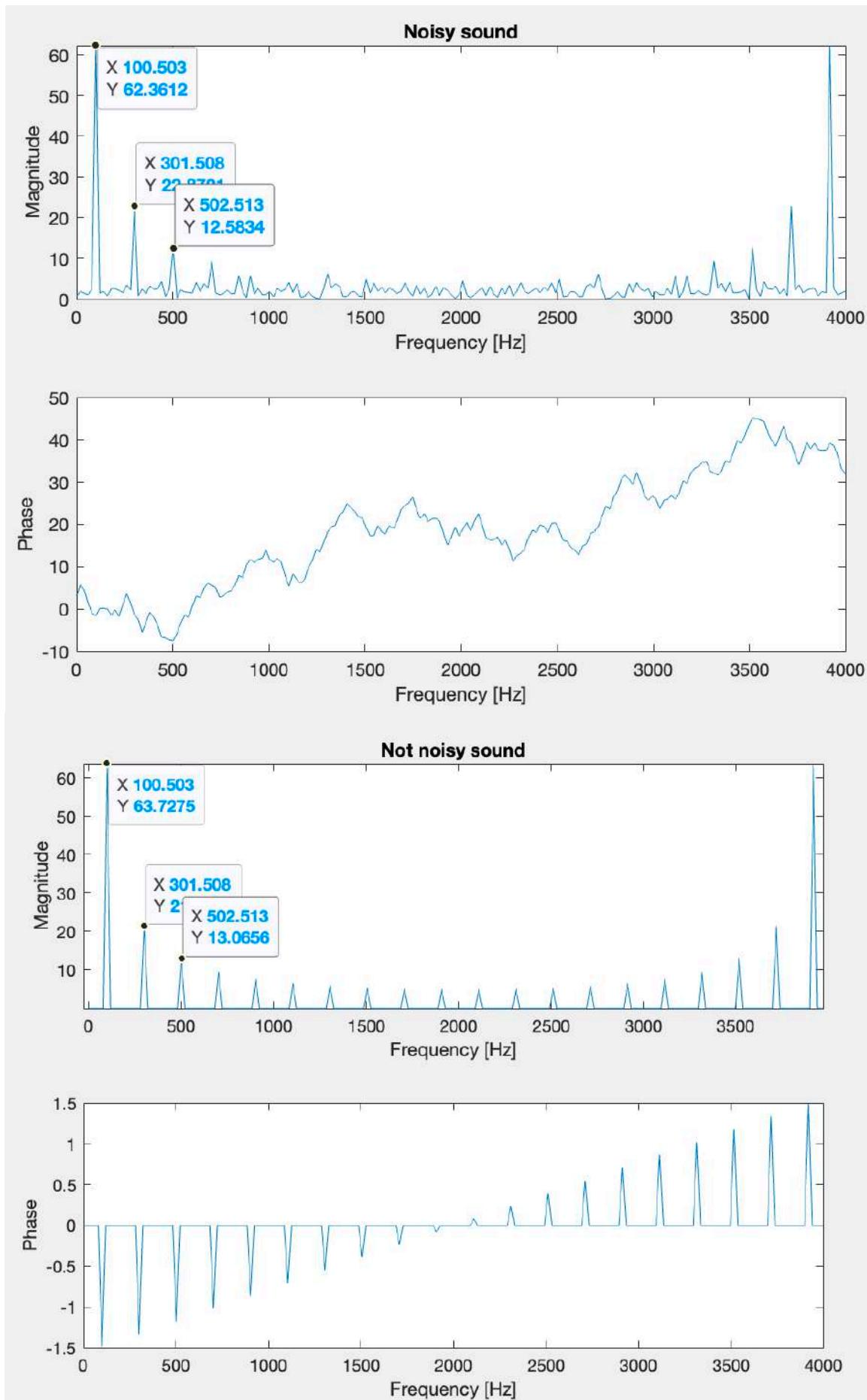
```
%% Compute the DFT of the signal
% We examine the frequency spectrum of the signal, plot the magnitude and
% the phase of the spectrum in one figure for each signal
fAx = 0: Fs/(signalLen - 1):Fs;
```

```
X = fft(x);
XandNOISE = fft(xandnoise);

figure();
subplot(2,1,1);
plot(fAx, abs(X));
xlabel('Frequency [Hz]'), ylabel('Magnitude');
subplot(2,1,2);
plot(fAx, unwrap(angle(X))); %unwrap make the shift phase
xlabel('Frequency [Hz]'), ylabel('Phase');

figure();
subplot(2,1,1);
plot(fAx, abs(XandNOISE));
xlabel('Frequency [Hz]'), ylabel('Magnitude');
subplot(2,1,2);
plot(fAx, unwrap(angle(XandNOISE))); %unwrap make the shift phase
xlabel('Frequency [Hz]'), ylabel('Phase');

% What are the differences between the two spectrum? What is the effect of
% noise. Can we identify a structure in the magnitude spectrum?
```

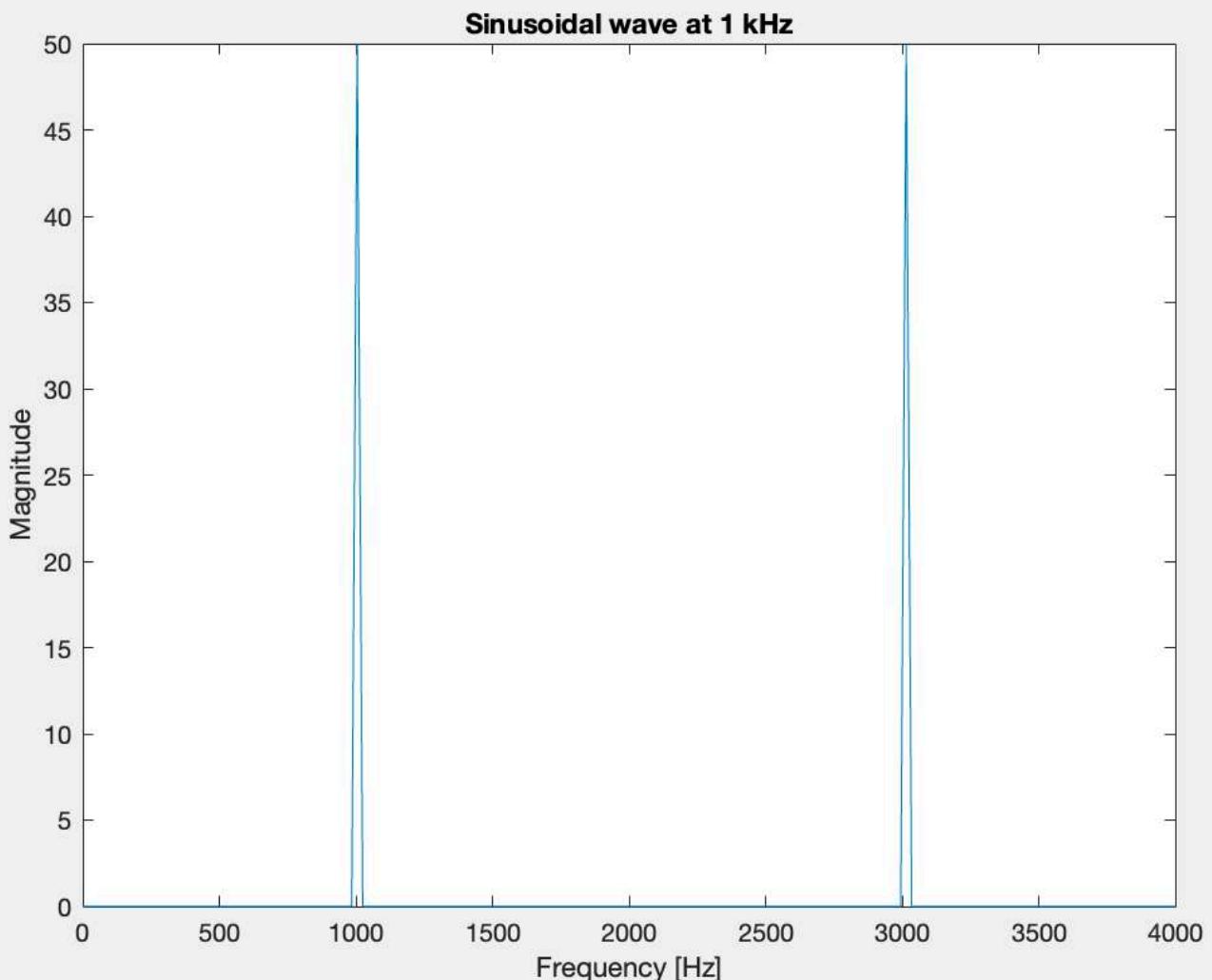


What are the differences between the two spectrum? What is the effect of noise. Can we identify a structure in the magnitude spectrum?

*Noise level is not so high, so we can see just the first harmonics contributes but its eviden in phase.*

Now analyse the signal of a square wave at 1kHz. Compute the DFT and inspect the spectrum. How many harmonics are presents?

```
%% Now analyze the signal of a square wave at 1kHz.
% Compute the DFT and inspect the spectrum. How many harmonics are
% presents?
f = 1000;
x = A * sin(2 * pi * f * t);
X = fft(x);
figure()
plot(fAx, abs(X));
xlabel('Frequency [Hz]'), ylabel('Magnitude')
title('Sinusoidal wave at 1 kHz')
```



*Ofcourse, only one.*

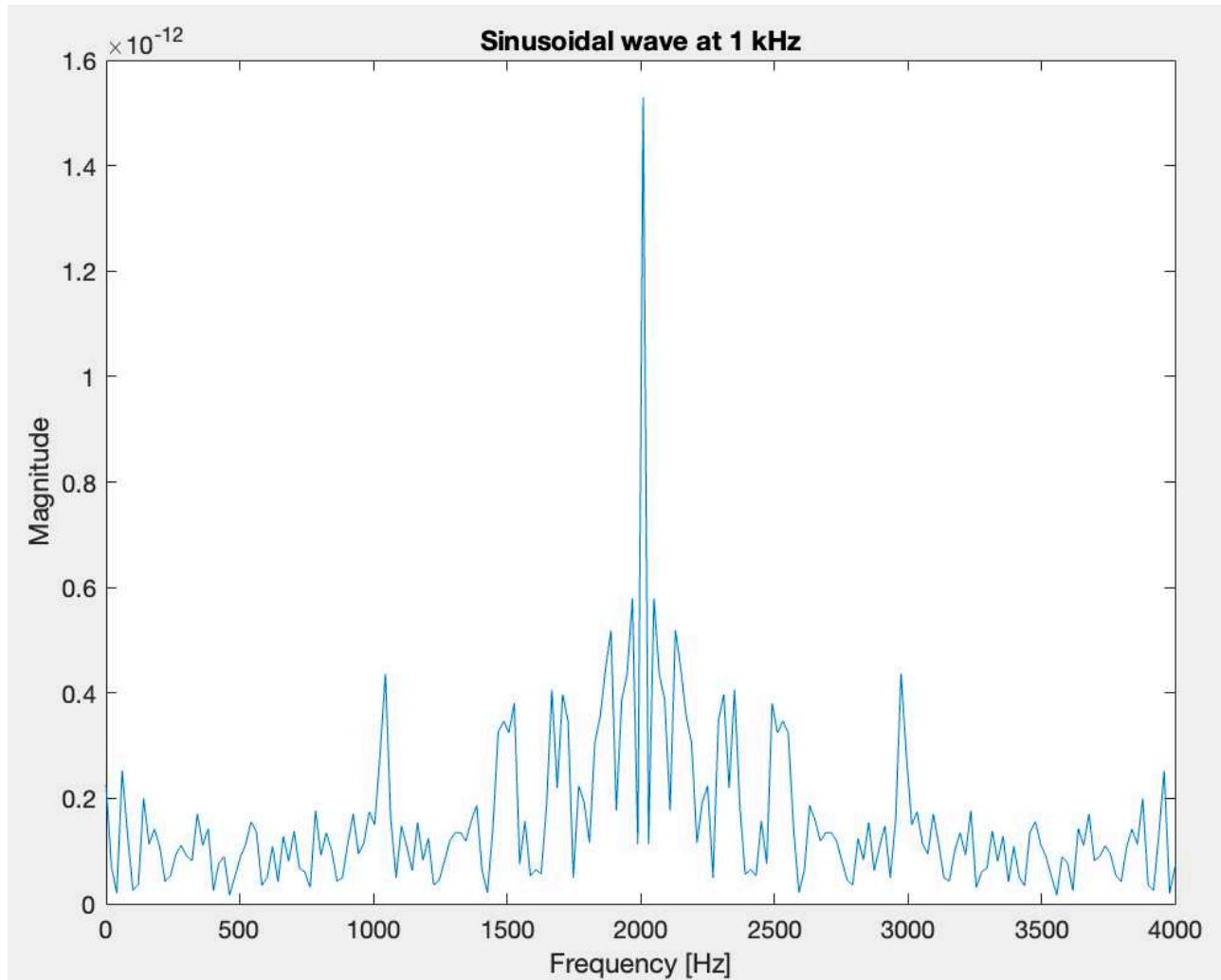
Change the sampling frequency to 2000Hz. What would you expect to see in the spectrum?

```
%% Now analyze the signal of a square wave at 1kHz.
% Compute the DFT and inspect the spectrum. How many harmonics are
% presents?
```

```

f = 2000;
x = A * sin(2 * pi * f * t);
X = fft(x);
figure()
plot(fAx, abs(X));
xlabel('Frequency [Hz]'), ylabel('Magnitude')
title('Sinusoidal wave at 1 kHz')

```



*The spectrum is basically destroyed because 2 KHz is exactly the Nyquist Frequency. Anyway, it is almost correct since we have the maximum around 2Khz.*

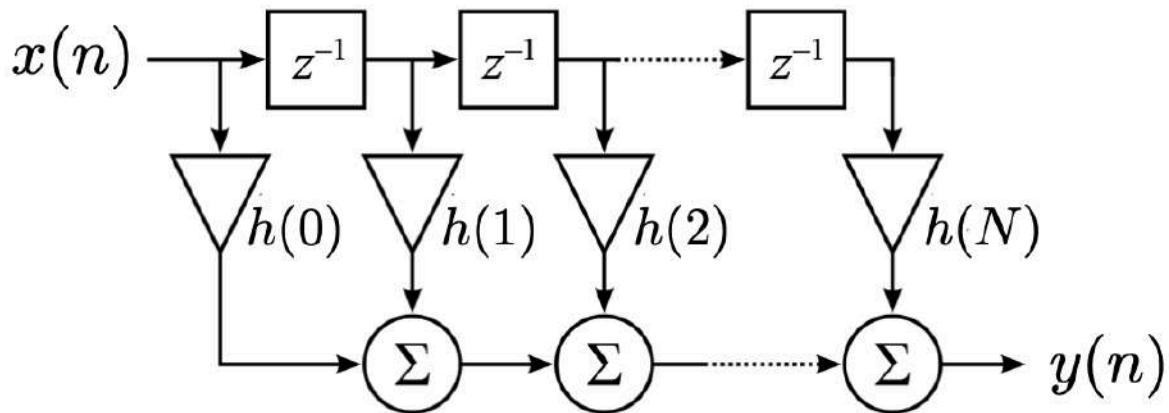
## Overlap and add

### Filtering in the frequency domain: overlap and add

`Filter` is an object that applies a mathematical operation on a signal put as input. We will focus on LTI filters since are mathematically nice and represent most of the acoustic filters.

We know that an LTI filter can be perfectly characterised by its impulse response. How can we find it? By putting as input a Dirac pulse.

In order to find the output of an LTI filter we have to compute the convolution between its impulse response and the signal input. The implementation of the operation is the following:



$$y(n) = \sum_{i=0}^N h(i)x(n-i)$$

By using this formula, implemented on MATLAB by the function

`y = conv(x, h);`

we have a complexity equal to  $O(N^2)$  that, practically speaking, is very high since the filter we use for acoustics purpose are extremely long (high value of N).

For example, the nominal “integration time” of the ear, defined here as the reciprocal of a critical bandwidth of hearing, is around 10ms below 500 Hz. (Some would say it is even longer.) At a 50 kHz sampling rate, this is  $N = 500$  samples. Therefore, FIR filters shorter than the ear’s “integration time” can easily be hundreds of taps long. FIR filters shorter than the ear’s integration time can generally be characterized by their magnitude frequency response (no perceivable “delay effects”). Since complexity of directly implementing the convolution is  $O(N^2)$ , computational issues may arise!

In order to solve this problem, we can rely on the convolution theorem

$$y(n) = x(n) \circledast h(n) = \sum_{m=0}^{N-1} x(m)h(n-m) \xrightarrow{\mathcal{F}\{\cdot\}} Y(\omega_k) = X(\omega_k)H(\omega_k)$$

and on the fast implementation of the DFT (FFT).

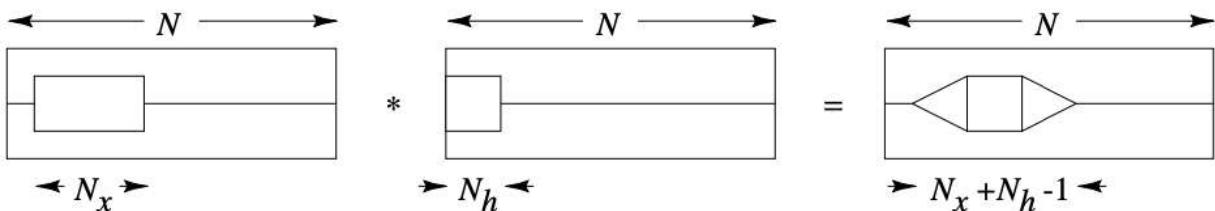
We exploit the theorem to speed-up the computation since by computing the FFT instead of the standard DTFT of both  $x$  and  $h$ , we reduce the computational complexity to  $O(N \log N)$ . To this number, we have to sum the complexity of the products  $O(N)$  and the one of the IFFT  $O(N \log N)$

N	FFT convolution	Time-domain convolution
4	176	16
32	2560	1024
64	5888	4096
128	13312	16384
256	29696	65536
512	311296	4194304

We remind that we work with digital signals. A sampling operation in time domain, corresponds to a periodisation in frequency domain. For this reason, we are interested in the evaluation of the acyclic convolution instead of the cyclic one. Since we want to obtain to get the same results as in acyclic convolution, we must add enough trailing zeros to the signals being convolved. If we don't add enough zeros, some of our convolution terms "wrap around" and add back upon others (due to modulo indexing). This can be thought of as time domain aliasing.

How many zeros do we have to add?

We know that the length of a convolution is



So the output of the filter has to respect this length. Adding zeros in the time domain results in more samples (closer spacing) in the frequency domain. This can be thought of as a higher "sampling rate" in the frequency domain. If we have a high enough sampling rate, we can avoid time domain aliasing. To fully exploit the efficiency of the FFT algorithm, the signals should be zero-padded to obtain a power of 2 as final length. (We remind that the benefits of the FFT are presents only if the number of samples is a power of 2).

Let us see these concepts applied in an exercise:

### Exercise 1 - Low Pass Filtering

Problem statement: design and implement a low-pass filter that has:

- $F_{cut} = 600\text{Hz}$
- Filter length  $L = 257 \text{ taps}$
- Signal  $x(n)$  is a sum of sinusoidal components (440, 880, 1000, 2000 Hz)
- We'll filter a single frame of length  $M = 512$  in the frequency domain.

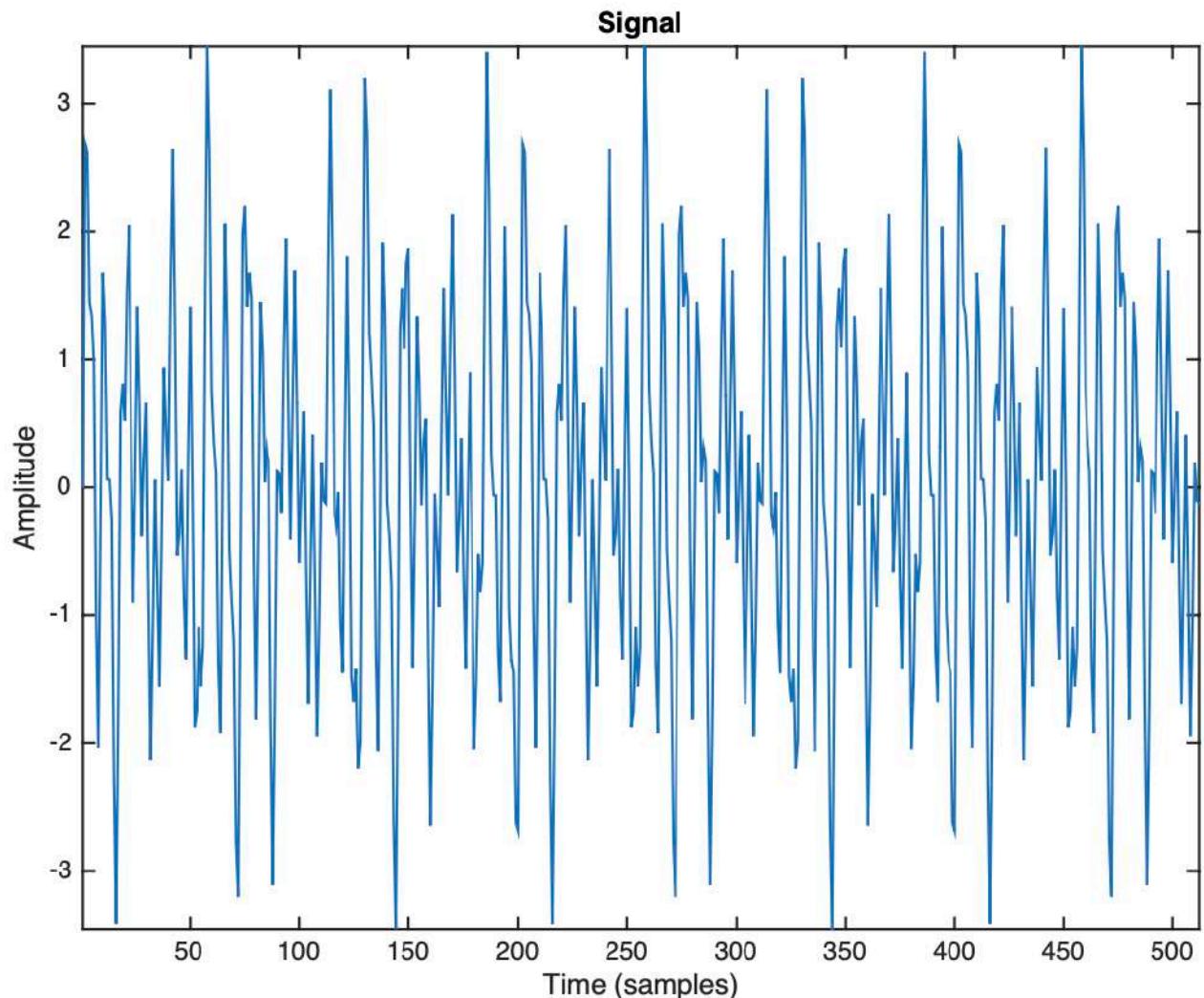
```
% *** signal parameters
Fs = 4000;
f = [ 440 880 1000 2000 ]; % frequencies
M = 512; % signal length
% *** generate a signal by adding up sinusoids
x = zeros(1,M);
n = 0:(M-1);
```

```

for k = 1:length(f);
    x = x + sin(2*pi*n*f(k)/Fs);
end

```

This is what we obtain



We now design the filter using the window method (so we trunk a sinc):

$$h(n) = 2B \operatorname{sinc}(2\pi Bn) = 2B \frac{\sin(2\pi Bn)}{2\pi Bn} = \frac{\sin(2\pi Bn)}{2\pi Bn}, \quad B = \frac{f_c}{f_s}$$

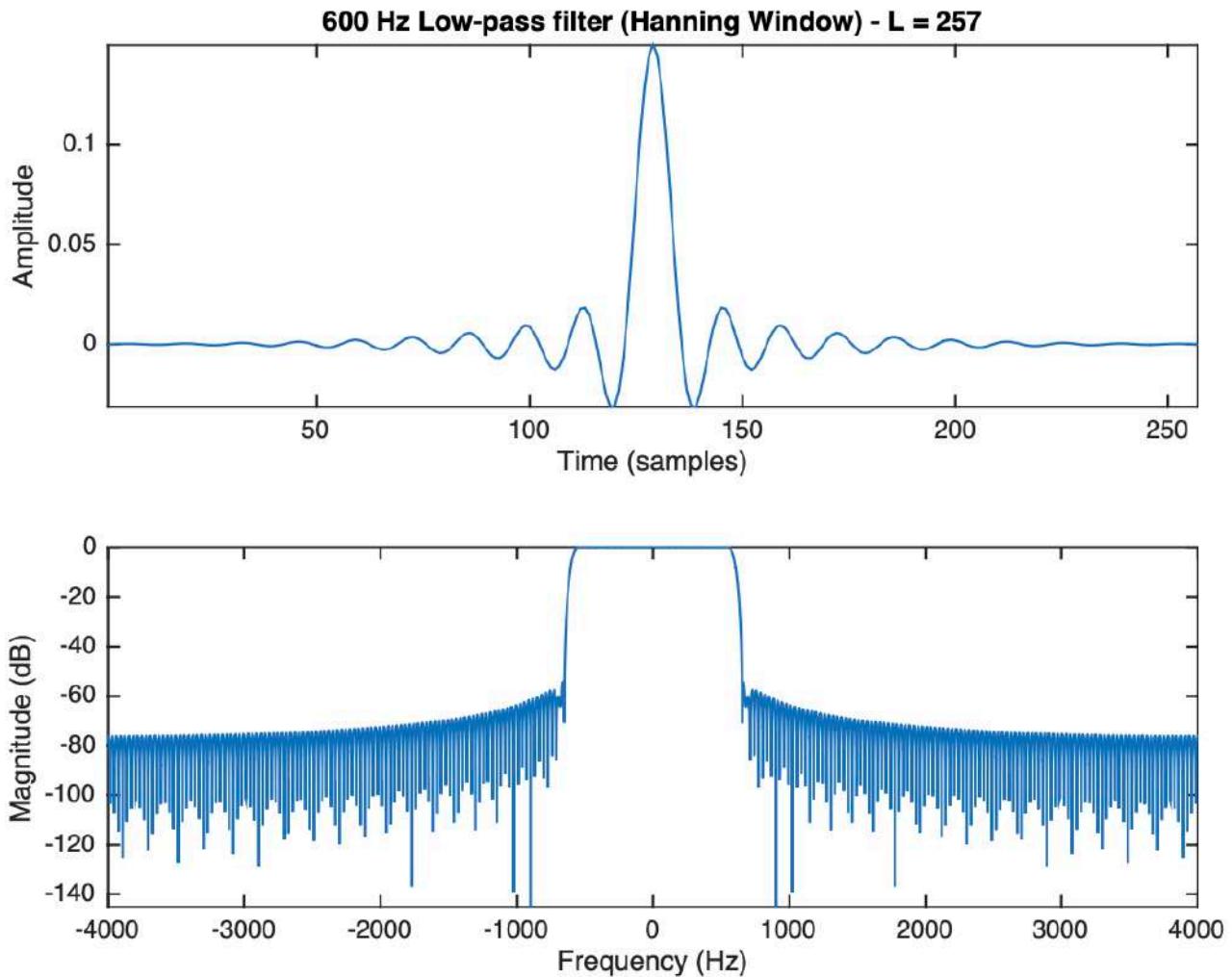
```

% *** filter parameters
L = 257; % filter length
fc = 600; % cutoff frequency
% *** design the filter using the window method
epsilon = .0001; nfilter = (- (L-1)/2 : (L-1)/2) + epsilon;
hideal = sin(2*pi*fc*nfilter/Fs) ./ (pi*nfilter);
h = hamming(L) .* hiddeal;

```

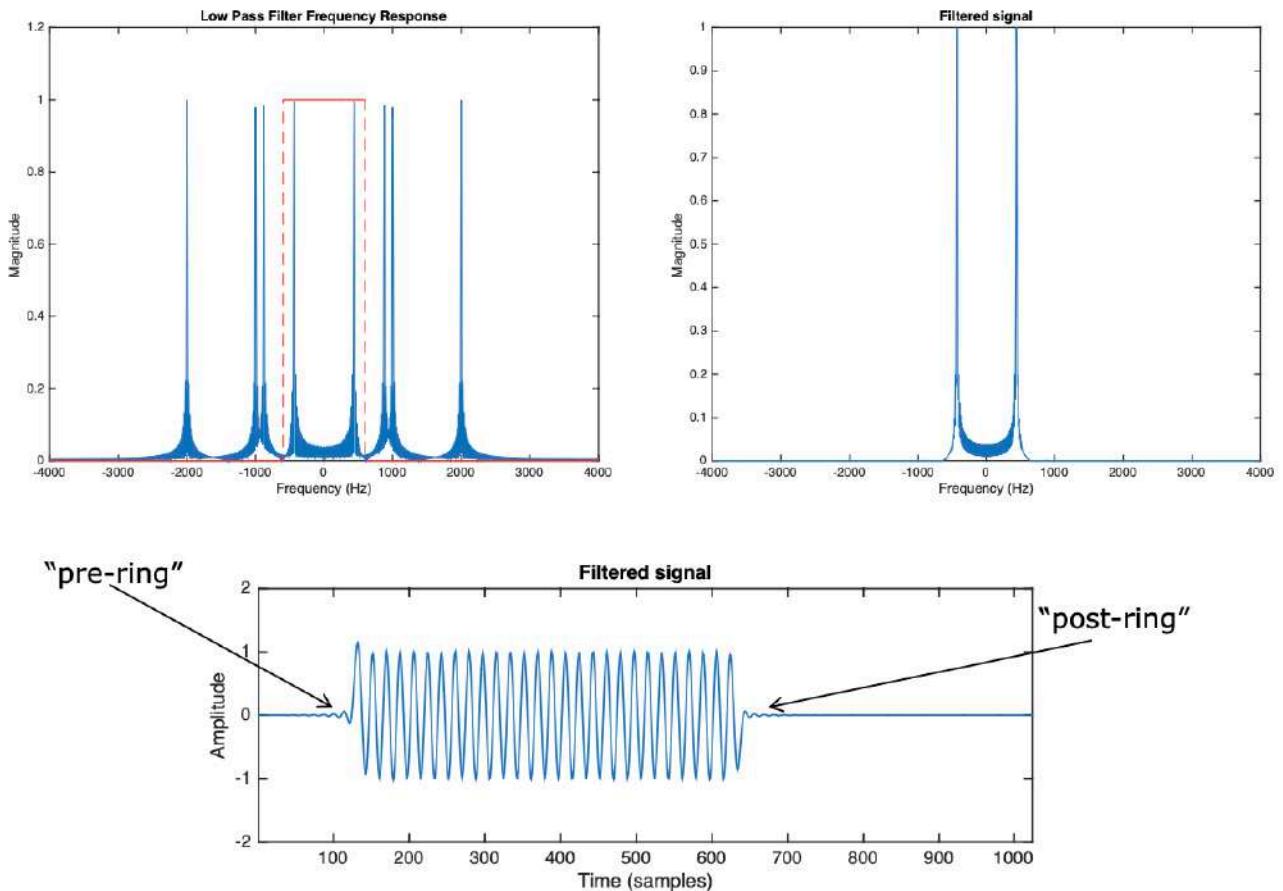
Remark: applying a Hamming window to the truncated sinc reduces the ringing effects due to the truncation.

This is what we obtain



We apply the convolution:

```
% *** choose the next power of 2 greater than L+M-1
Nfft = 2^(ceil(log2( L + M -1)));
% Transform the signal and the filter and zero pad
S = fft(sigzp,Nfft);
H = fft(hzp,Nfft);
% perform frequency domain multiplication...
Sfilt = S.*H;
% ...and obtain the filtered signal through ifft
sfilt = ifft(Sfilt);
```



As we can see, some delay has been added.

### Exercise 2 – Time domain aliasing (or why it is important to fulfil the condition of the length of the convolution)

This example shows the effect of insufficient zero padding (it can also be thought of as under-sampling in the frequency domain).

Problem statement: implement a low-pass filter:

- Filter length  $L = 65$  samples
- Data frame length  $M = 100$ ; signal composed of two impulses:
  - At time 0
  - At time  $M - \frac{L-1}{4} = 86$
- Try different FFT lengths:
  - $N = M$
  - $N = M + \frac{L}{4}$
  - $N = M + \frac{L}{2}$
  - $N = M + L$
- We remind that the anti-aliasing condition is given by:  $N > M + L - 1$

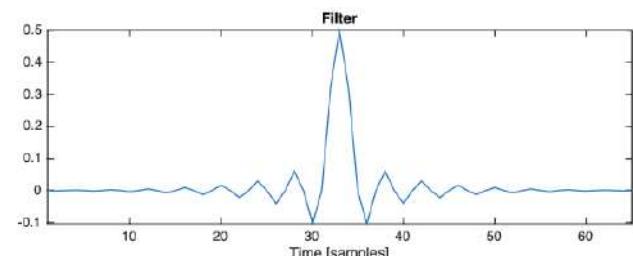
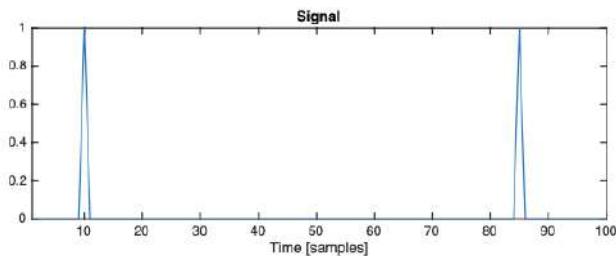
We implement signal and filters

```
M = 100; % signal length
L = 65; % filter length
```

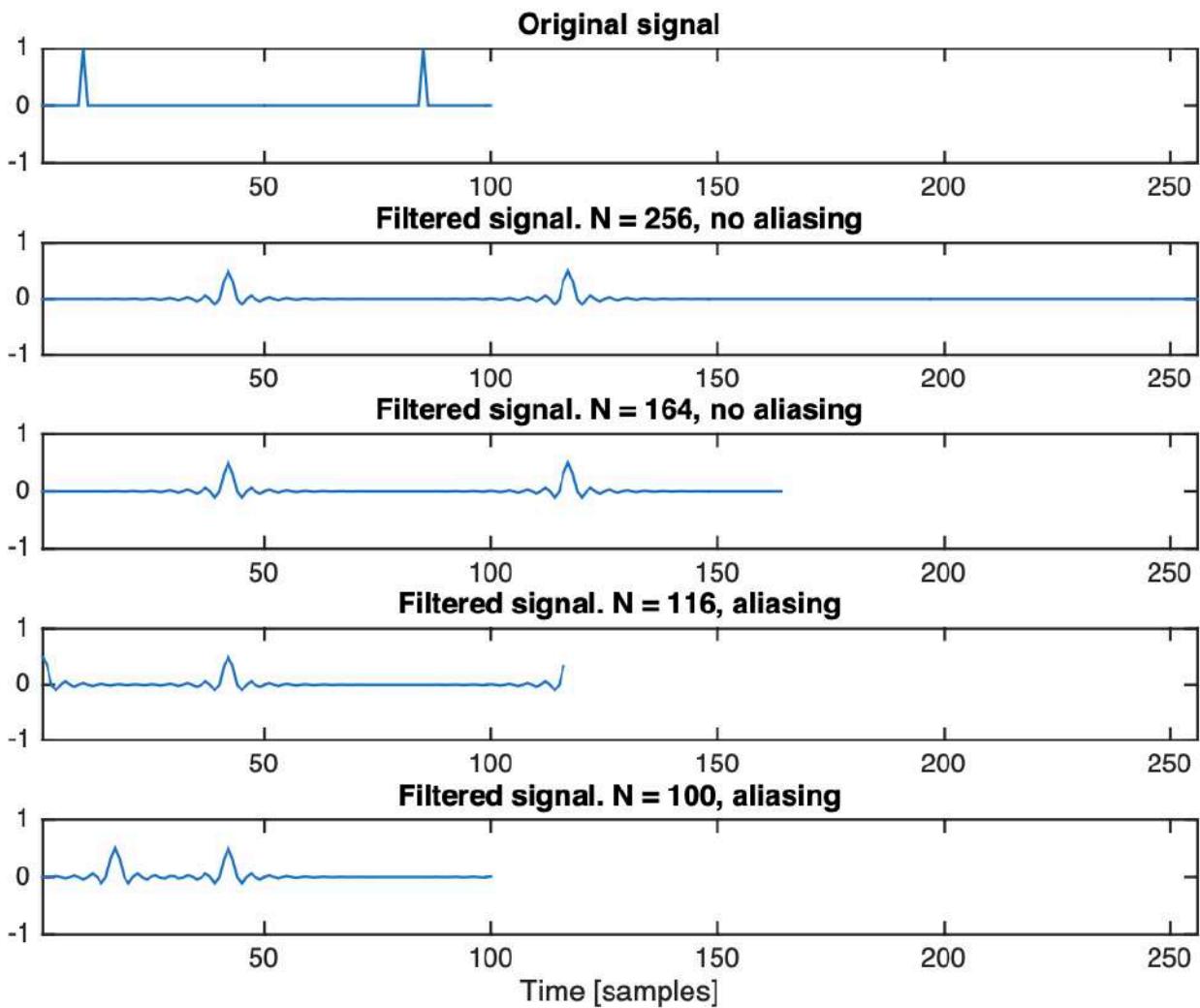
```
% define the signal
x = zeros(M,1); x(10) = 1; x(86) = 1;
% define the filter
h = fir1(L-1,0.5).'';

% zero padding
N = 100; % try also N = 116, N = 132, N = 164, N = 256

x0 = [x; zeros(N-M,1)];
h0 = [h; zeros(N-L,1)];
X0 = fft(x0); H0 = fft(h0);
y0 = ifft(X0.*H0);
```



These are the results we obtain:



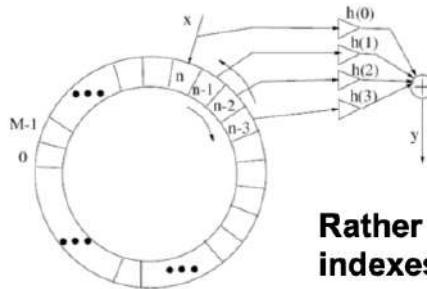
## Filtering of indefinite length signals

Sometimes it is not possible to have the signals in their entire length to apply the convolution. An example is offered by the live events where the signal is basically created second by second.

There are two approaches that we can exploit to solve this issue: the first one is by using a circular buffer and the second one by using overlap and add.

### Circular Buffer

Theoretically, there is no problem doing this with direct convolution. Since  $h$  is finite in length, we only need to store the past  $N_h - 1$  samples of the input signal  $x$  to calculate the next output. An efficient and widely adopted approach is the «circular buffer». This strategy is useful when we have limited computational power and memory. It is surely a problem when the filter is very long.



**Rather than moving data, move the indexes that access the data**

### Overlap & Add:

The real approach is the one made with Overlap and Add. We have a signal that we divide in multiple frames and we process one frame at a time. We do not need the entire signal in advance! We process the signal step by step solving the problem of knowing the length of the filter in advantage. So let us see how we proceed:

1. We chop up the input signal  $x$  by windowing it

$$x_m(n) \triangleq x(n)w(n - mR) \text{ with } n \in (-\infty; +\infty)$$

2. We compute the cyclical convolution between the filter and the portion;
3. We sum the results of the single convolutions together.

We have to verify that for this frame-by-frame spectral processing to work, we are able to reconstruct  $x$  from the individual overlapping frames. This can be written as

$$x(n) = \sum_{m=-\infty}^{\infty} x_m(n) \quad x_m(n) \triangleq x(n)w(n - mR) = x(n) \sum_{m=-\infty}^{\infty} w(n - mR)$$

It follows that:

$x(n) = \sum_{m=-\infty}^{\infty} x_m(n)$  if and only if  $\sum_{m=-\infty}^{\infty} w(n - mR) = 1$ . Condition that is called *constant overlap-and-add (COLA) condition on the analysis window w*.

This condition is not written in stone: we can be more elastic and relax this condition in some situations. But we can surely affirm that all windows which obey the constant-overlap-add constraint will yield perfect reconstruction of the original signal from the data frames by overlap-add. There is no constraint on window type, only that the window overlap-adds to a constant for the hop size used.

Examples:

- Rectangular window at 0% overlap (Hop size R = Window size M)
- Rectangular window at 50% overlap ( $R = \frac{M}{2}$ )
- Bartlett window at 50% overlap ( $R = \frac{M}{2}$ )
- § Hamming window at 50% overlap ( $R = \frac{M}{2}$ )

- § Hamming window at 25% overlap ( $R = \frac{M}{4}$ )
- Any window with R=1 (“sliding FFT”)

Let us say that we want to perform filtering by using overlap and add framework. What we would do in time domain is

$$y(n) = x(n) * h(n) = \sum_{m=-\infty}^{+\infty} x_m(n) * h(n)$$

Where we consider a discrete infinite function for which  $-\infty < n < \infty$

We substitute the condition into the  $Y(\omega)$  definition to find the result for every frame by starting from the DTFT.

$$\begin{aligned} Y(\omega) &= \sum_{n=-\infty}^{+\infty} y(n) e^{-j\omega n} \\ &\stackrel{y(n)=\sum_{m=-\infty}^{+\infty} x_m(n)*h(n)}{=} \sum_{n=-\infty}^{+\infty} \sum_{m=-\infty}^{+\infty} x_m(n) * h(n) e^{-j\omega n} = \\ &= \sum_{m=-\infty}^{+\infty} \left[ \sum_{n=-\infty}^{+\infty} x_m(n) * h(n) \right] e^{-j\omega n} = \sum_{m=-\infty}^{+\infty} X_m(\omega) H(\omega) \end{aligned}$$

Up to now we have considered continuous frequency domain (discrete signal  $\rightarrow$  continuous frequency but periodic). But we work with digital systems! Thus, we cannot have continuous frequency response. So we have to sample the continuous frequency response in order to make it discrete. We know that the length of the convolution will be length of the filter + length of the frame of the signal.

In fact, we observe that the filter, the window, and the extracted frames have finite support.

$$\begin{aligned} h(n) &\neq 0 \quad \text{for } n \in [0, N_h - 1] \\ w(n) &\neq 0 \quad \text{for } n \in [0, M - 1] \\ x_m(n) &\neq 0 \quad \text{for } n \in [mR, mR + M - 1] \end{aligned}$$

As a consequence, the convolution of m-th frames with the filter has finite support, with  $N_h + M - 1$  non-zero terms:

$$y_m(n) = x_m(n) * h(n) \neq 0 \quad \text{for } n \in [mR, mR + N_h + M - 2]$$

So, we use the FFT to compute the convolution, considering only the non-zero samples. For convenience, we shift back each frame to the origin by defining

$$\tilde{x}_m(n) \triangleq x_m(n + mR)$$

Convolving it with the filter, we obtain

$$\tilde{y}_m(n) = \tilde{x}_m(n) * h(n) = y_m(n + mR)$$

We notice that  $\tilde{y}_m(n)$  is time-limited to  $L = N_h + M - 1$  non-zero samples. In other words,  $\tilde{y}_m(n)$  is “band-limited in the time domain”, thus its continuous DTFT  $Y_m(\omega)$  can be critically sampled at the points

$$\omega_k = \frac{2k\pi}{L}, \quad \text{for } k = 0, 1, 2, \dots, L - 1$$

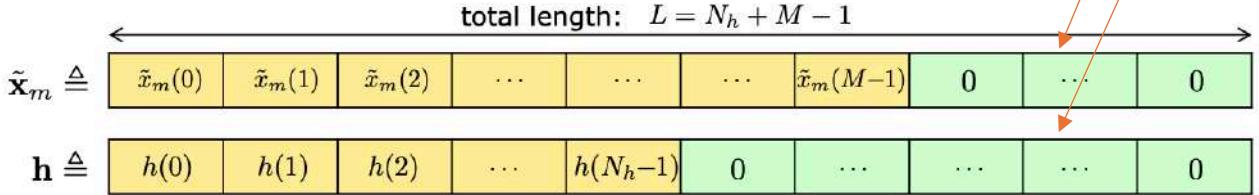
We are now ready to compute the  $\tilde{Y}_m(\omega_k)$  that corresponds to the length  $L$  Discrete Fourier Transform (DFT) of the sequence  $[\tilde{y}_m(0), \tilde{y}_m(1), \dots, \tilde{y}_m(L - 1)]$ . Clearly,  $\tilde{Y}_m(\omega_k)$  can be computed as

$$\tilde{Y}_m(\omega_k) = \tilde{X}_m(\omega_k)H(\omega_k)$$

Where:

- $\tilde{X}_m(\omega_k)$  and  $H(\omega_k)$  are the sampled DTFTs of  $\tilde{x}_m(n)$  and  $h(n)$
- They can be computed via DFT

We implement the algorithm by introducing zero padding.



$$\begin{cases} \tilde{\mathbf{X}}_m \triangleq \text{FFT}(\tilde{\mathbf{x}}_m) \\ \mathbf{H} \triangleq \text{FFT}(\mathbf{h}) \end{cases} \quad \xrightarrow{\hspace{1cm}} \quad \tilde{\mathbf{Y}}_m = \tilde{\mathbf{X}}_m \otimes \mathbf{H}$$

The FFT of the filtered frame samples is obtained as the element-wise product of the two vectors

The vector  $\tilde{\mathbf{Y}}_m$  contains the samples of  $\tilde{Y}_m(\omega_k)$ , i.e.

$$\tilde{\mathbf{Y}}_m \triangleq \begin{bmatrix} \tilde{Y}(\omega_0) & \tilde{Y}(\omega_1) & \tilde{Y}(\omega_2) & \cdots & \tilde{Y}(\omega_{L-1}) \end{bmatrix}$$

Taking the inverse DFT, we obtain the vector  $\tilde{\mathbf{y}}_m = \text{IFFT}(\tilde{\mathbf{Y}}_m)$ , which contains the samples of the filtered frame:

$$\tilde{\mathbf{y}}_m \triangleq \begin{bmatrix} \tilde{y}_m(0) & \tilde{y}_m(1) & \tilde{y}_m(2) & \cdots & \tilde{y}_m(L-1) \end{bmatrix}$$

The FFT convolution is operated for every frame extracted from the input signal.

Again a remark, in order to exploit the FFT implementation, we have to add a number of zeros in a way that the length is equal to a power of two.

We have to remind also that each frame has been shifted back by  $mR$  samples: defining, for simplicity, the SHIFT operator, we have

$$\tilde{x}_m(n) = \text{SHIFT}_{-mR}[x_m(n)] = x_m(n + mR)$$

After computing the convolution, we need to shift the result to the original position, namely

$$y_m(n) = \text{SHIFT}_{mR}[\tilde{y}_m(n)]$$

Putting everything together we obtain:

$$y(n) = \sum_{m=-\infty}^{\infty} y_m(n) \underset{y_m(n)=\text{SHIFT}_{mR}[\tilde{y}_m(n)]}{=} \sum_{m=-\infty}^{\infty} \text{SHIFT}_{mR}[\tilde{y}_m(n)] =$$

$$= \sum_{m=-\infty}^{\infty} SHIFT_{mR}[\tilde{x}_m(n) * h(n)] = \sum_{m=-\infty}^{\infty} SHIFT_{mR} [IFFT\{FFT\{\tilde{x}_m\} \otimes FFT\{h\}\}_{n-mR}]$$

The output  $y(n)$  is given by the infinite sum of the sinusoidal output which corresponds to the shifted results of the  $\tilde{y}_m(n)$ .  $\tilde{y}_m(n)$  is the result of the cyclical convolution between the  $x$  windowed and  $h$ . To perform it properly we have to satisfy the COLA and anti-aliasing condition.

### Exercise 3 - OLA Filtering

We use an input signal that is a train impulses and we filter it with a low-pass filter that has  $n = 32$  samples. The analysis window we use is

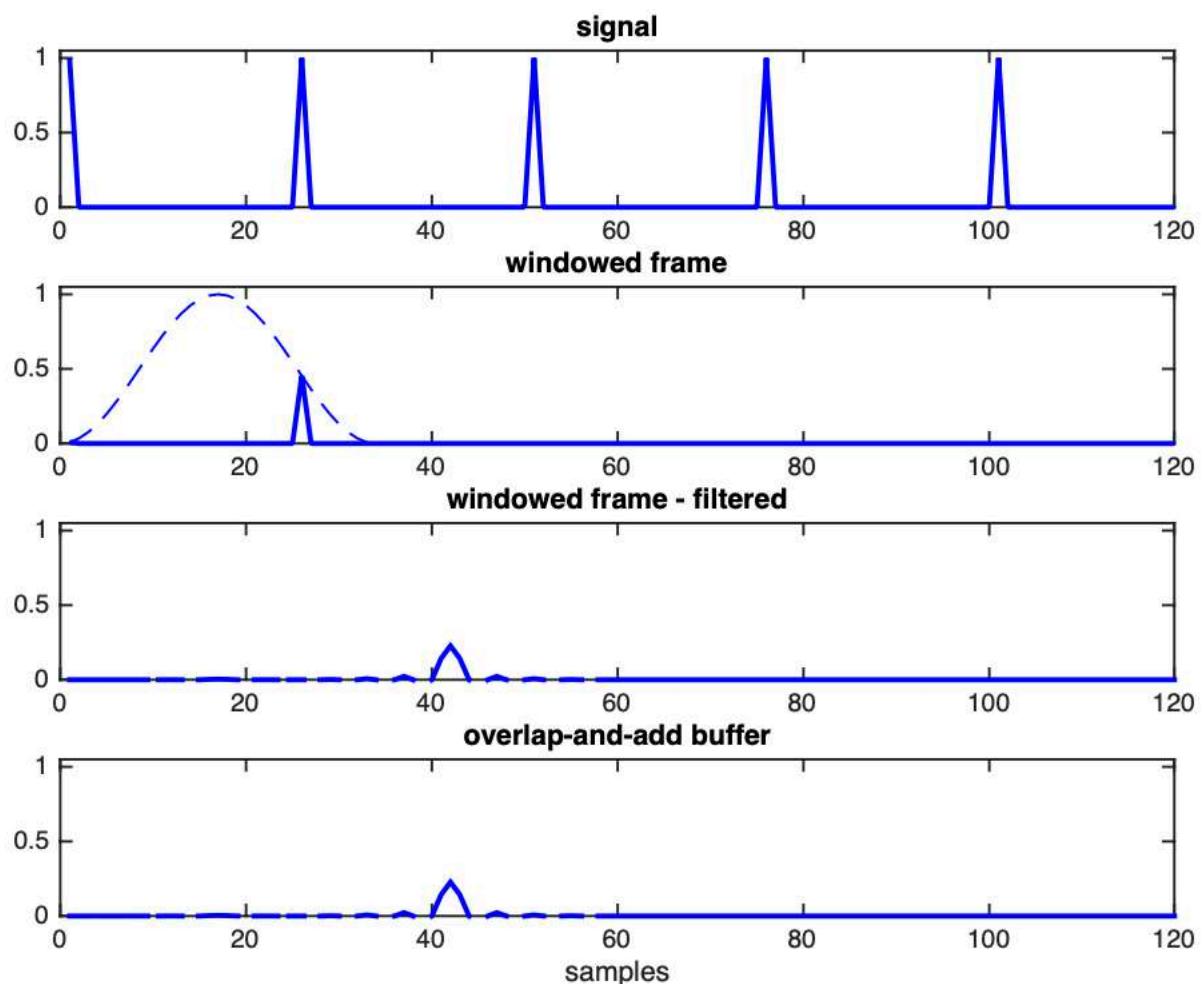
```
% define the input signal
L = 10000; % tot signal length
x = zeros(1,L);
x(1:25:L) = 1;
% design the filter
Nh = 33; % filter length
h = fir1(Nh-1, 0.5); % low pass filter
% design the analysis window
M = 33;
w = hanning(M)';
R = (M-1)/2+1;
% window length
% hanning window
% hop size
% compute the best value for the length of fft:
% Nfft >= M + Nh - 1, and Nfft is pow. of 2
Nfft = 2^(ceil(log2( M + Nh - 1)));
% compute the FFT of the filter
H = fft(h,Nfft);
% allocate the space for output signal
y = zeros(1,L+Nh-1);
% number of frames to be processed
nframes = floor( (L-M)/R ); % number of frames

% OLA algorithm
for m = 1:nframes
    % extract the m-th frame and apply the window
    m_idxs = (m-1)*R+1 : (m-1)*R+M;
    xm = x(m_idxs) .* w;
    % compute the FFT
    Xm = fft(xm,Nfft);
    % compute the convolution in freq. domain
    Ym = Xm.*H;
```

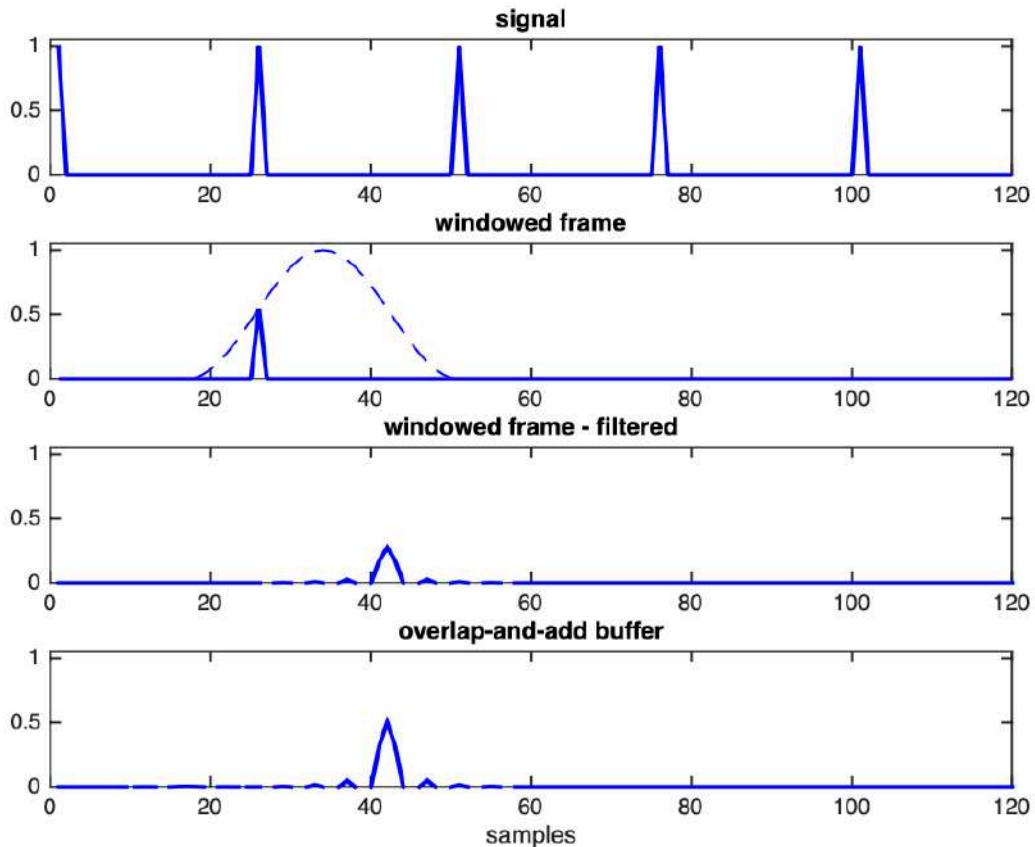
We set the FFT points in order to satisfy the length of the filter output and the power of two condition.

```
% come back to time domain
ym = real(ifft(Ym));
% extract only non-zero entries
ym = ym(1:M+Nh-1);
% overlap and add
ola_idx = (m-1)*R+1 : (m-1)*R + (M+Nh-1);
y(ola_idx) = y(ola_idx) + ym;
end
```

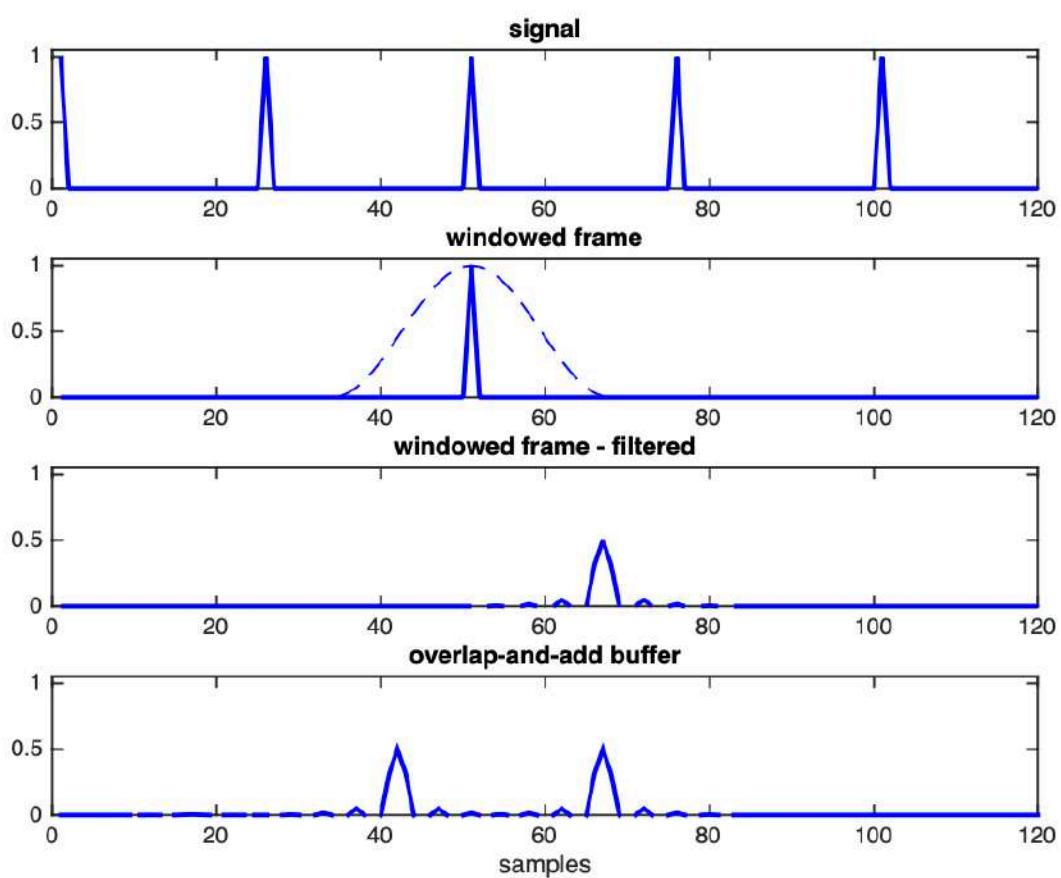
First position of the window:



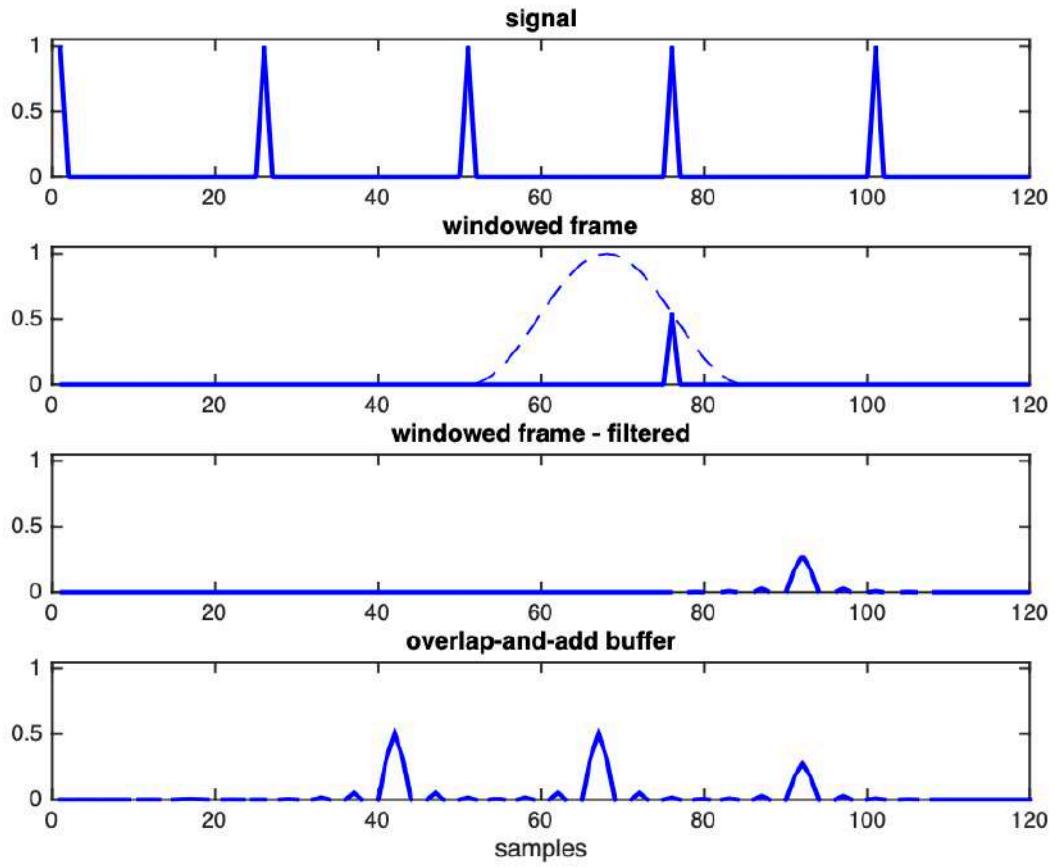
Second position of the window:



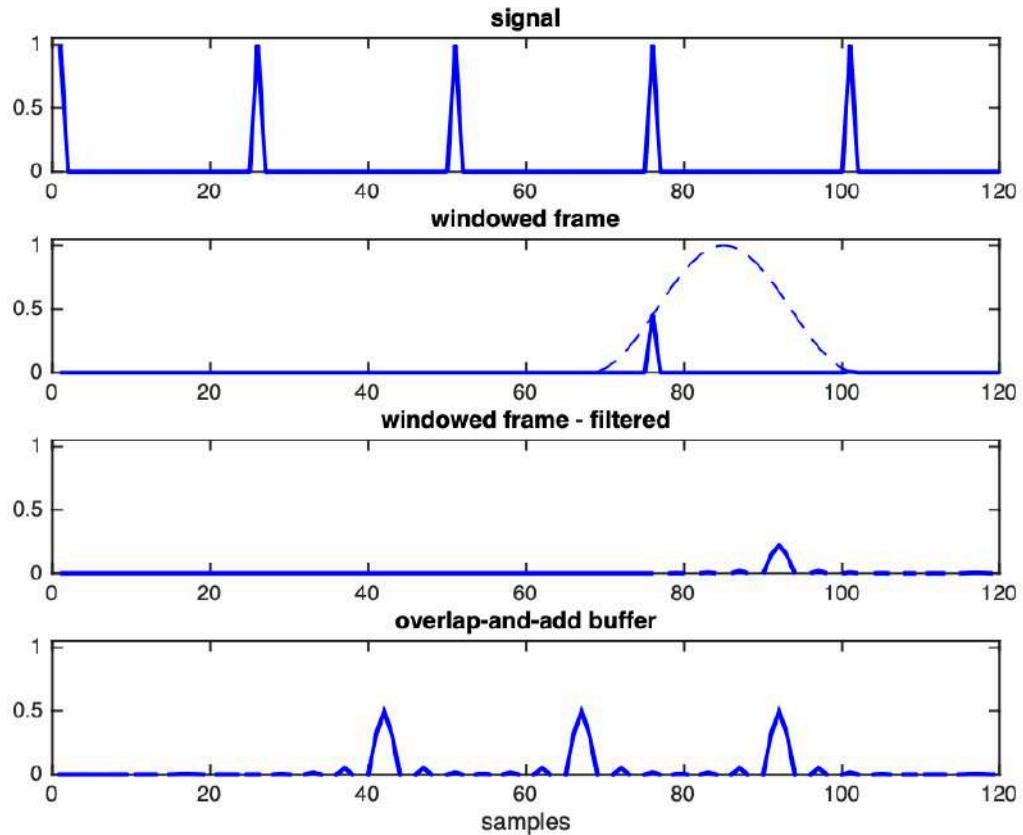
Third position:



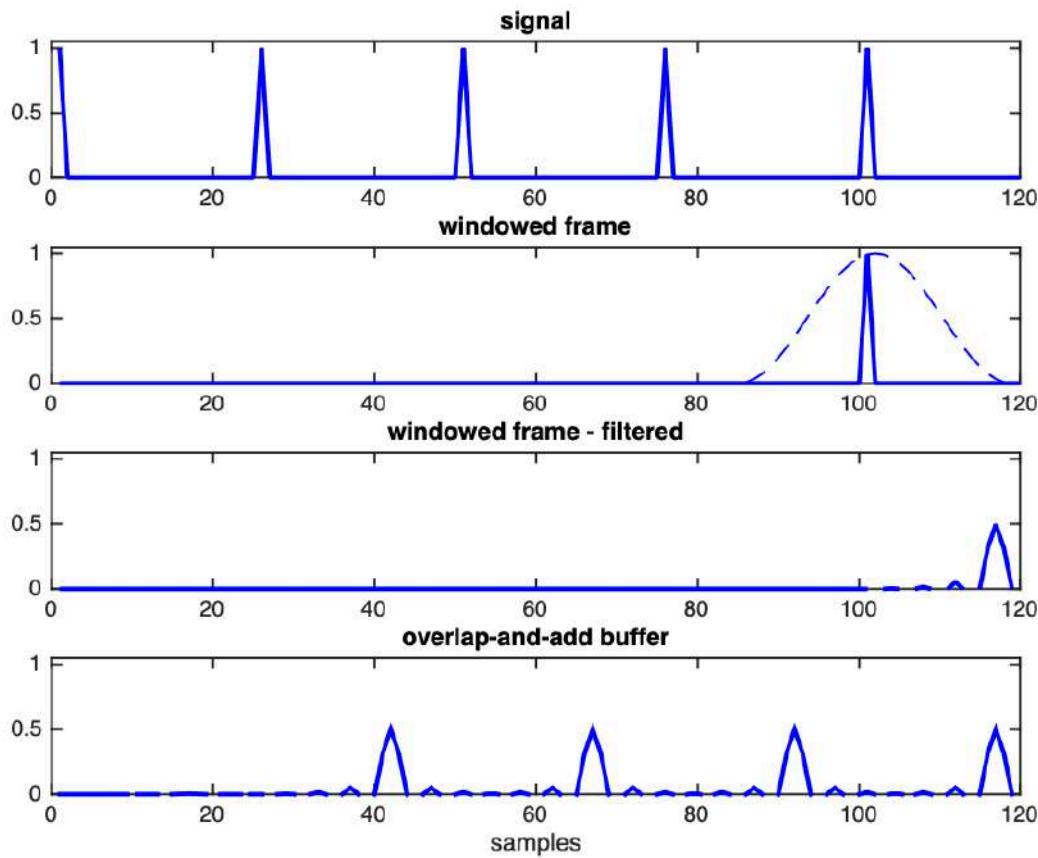
Fourth position:



Fifth position:



Sixth position:



And so on and so forth.

## Applications

This approach is widely used in live audio application especially to apply effects on the signals. It is applied for example in real-time convolution reverbs (e.g.

VST plugins)

- We specify the desired room impulse response (RIR), that can be very long (even more than 10000 samples at 44,1 kHz);
- We convolve the signal with the RIR and we obtain the output.

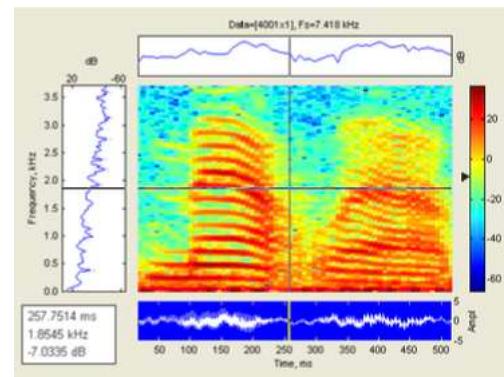
Another aspect for which this tool is perfectly fit, it is the fact that RIR can be time varying. So, it is better that the convolution is evaluated step-by-step.



Another implementation is given by the multichannel reproduction systems (e.g. wave field synthesis). In this type of acoustical systems,  $L$  loudspeakers contribute to synthesise a desired wavefield in a prescribed area. To do so, the source signal is properly filtered, in an independent manner for each loudspeaker. To make everything work, we need real time FIR filtering over  $L$  independent channels. The filter lengths can reach up to 4000 samples.

Another important application is the so-called Short Time Fourier Transform that is a 3D representation of an audio signal that basically combines the analysis in time domain with the one in frequency domain. It can be visualised as the following image (first axis (x): time, second axis (y): frequency, third axis (colour): amplitude).

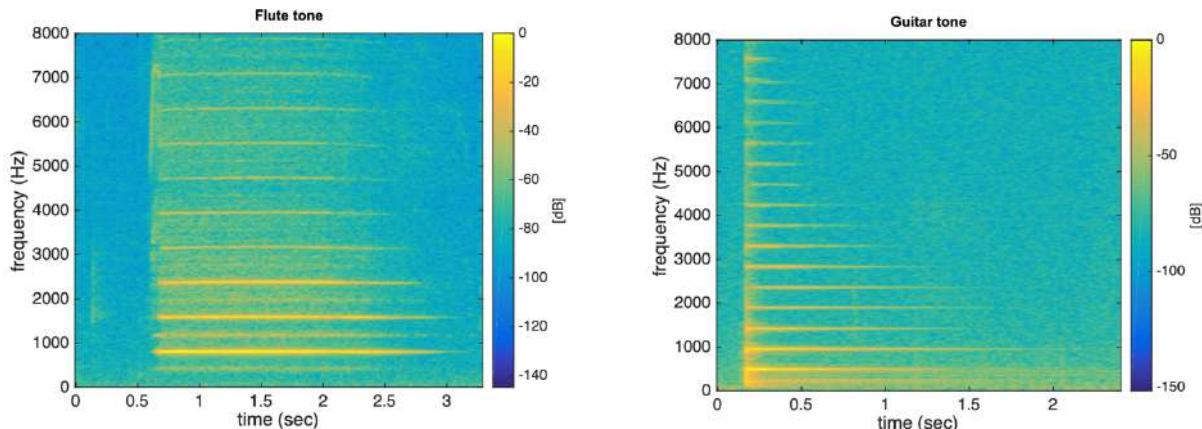
Just by looking at the pictures, we can imagine that we need to find a trade-off between the resolution in time and the resolution in frequency.



Let us see how to compute it on MATLAB.

```
[x Fs] = audioread('guitar.wav'); % read the source file
M = floor(0.050*Fs); %window length (50msec)
R = floor(M/4); %hop size
N = 2^(ceil(log2(M))); %power of 2 larger than M
w = hanning(M)'; %length M hanning window
Nframes = floor((length(x)-N)/R) + 1; %number of frames
% Compute the STFT
STFT = zeros(Nframes , N/2);
for m = 0 : Nframes
    xm = x(m*R+1:m*R+M) ;
    ym = w.*xm;
    temp = abs ( fft(ym, N) ) ;
    STFT(m+1, :) = temp (1:N/2) ;
end

STFT = fliplr(STFT)';
% Plot the SpectroGram
t = [ 0 : Nframes - 1]*R/Fs ;
f = Fs * [ 0 :N/2 - 1 ] /N;
imagesc(t,f,10*log10(abs(STFT).^2));
xlabel('time (sec)');
ylabel('frequency (Hz)');
```



## Exercise 1 - FIR Filtering Example

```
%%%%%
% Overlap and add
% DAAP course
% 2024
% Mirco Pezzoli
% Exercise 1
% %%%%%%
clc
clear
close all

%% FIR filtering example
% Generate a signal with a sawtooth and square wave. The sampling
% frequency is equal to 8kHz and the fundamental frequency is 500Hz and
% 1kHz. The signal length is 0.06s.

fs = 8e3;           % Sample frequency [Hz]
ts = 1/fs;          % Sample period [s]
t = 0:ts:0.06;      % Time axis [s]
f1 = 500;           % Frequency of the square wave [Hz]
f2 = 1e3;
xSawtooth = sawtooth(2*pi*f1*t);
xSquare = square(2*pi*f1*t);

% Plot the signals
figure()
subplot(2,1,1)
plot(t, xSawtooth);
xlabel('Time [s]'), title('Saw Tooth wave form')
subplot(2,1,2)
plot(t, xSquare);
xlabel('Time [s]'), title('Square wave form')

% Design an ideal low pass filter with cut off frequency 1.5kHz. The filter
% length in samples is 512. Smooth the ideal filter with an hamming window.
% (hint: doc hamming)

L = 512; % filter length
fc = 1.5e3; % cutoff frequency
% Design the filter using the window method
nfilt = (-(L-1)/2:(L-1)/2); % We center the sinc in the vector
hideal = sin(2*pi*fc*nfilt/fs) ./ (pi*nfilt);
h = hamming(L).* hideal;    % Smooth the sinc with hamming window

% Plot the filter in time and the magnitude of its spectrum in one window
H = fft(h, fs);
figure();
subplot(2,1,1);
plot(h); %without time axis (it is chosen automatically)
axis tight
xlabel('Time (samples)');
ylabel('Amplitude');
title('1500 Hz Low-pass filter (Hanning Window) - L = 512');

subplot(2,1,2);
f_ax = linspace(-fs/2,fs/2,fs);
plot(f_ax, 20*log10(abs(fftshift(H))));
```

```

xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');

% Filter the sawtooth and the square with the filters in time domain and
% plot the output of the filtering (use conv)

ySawtooth = conv(xSawtooth, h);
ySquare = conv(xSquare, h);

% or use filter

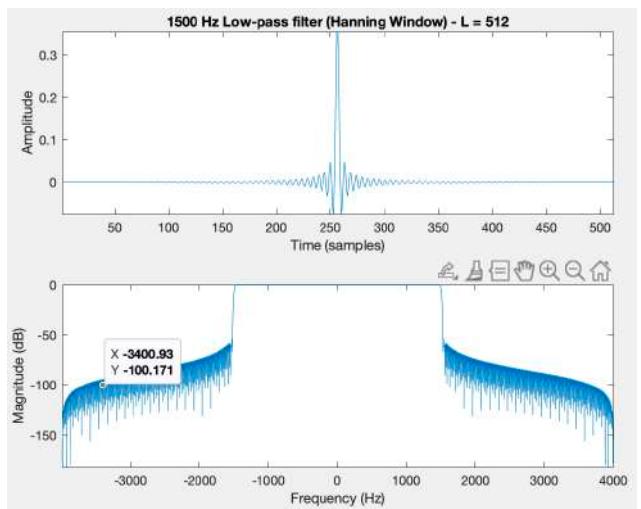
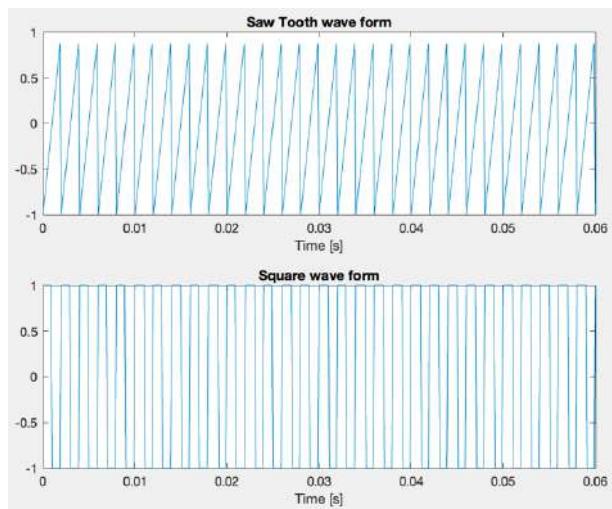
ySawtooth = filter(h, 1, xSawtooth); %1 is the denominator
ySquare = filter(h, 1, xSquare);

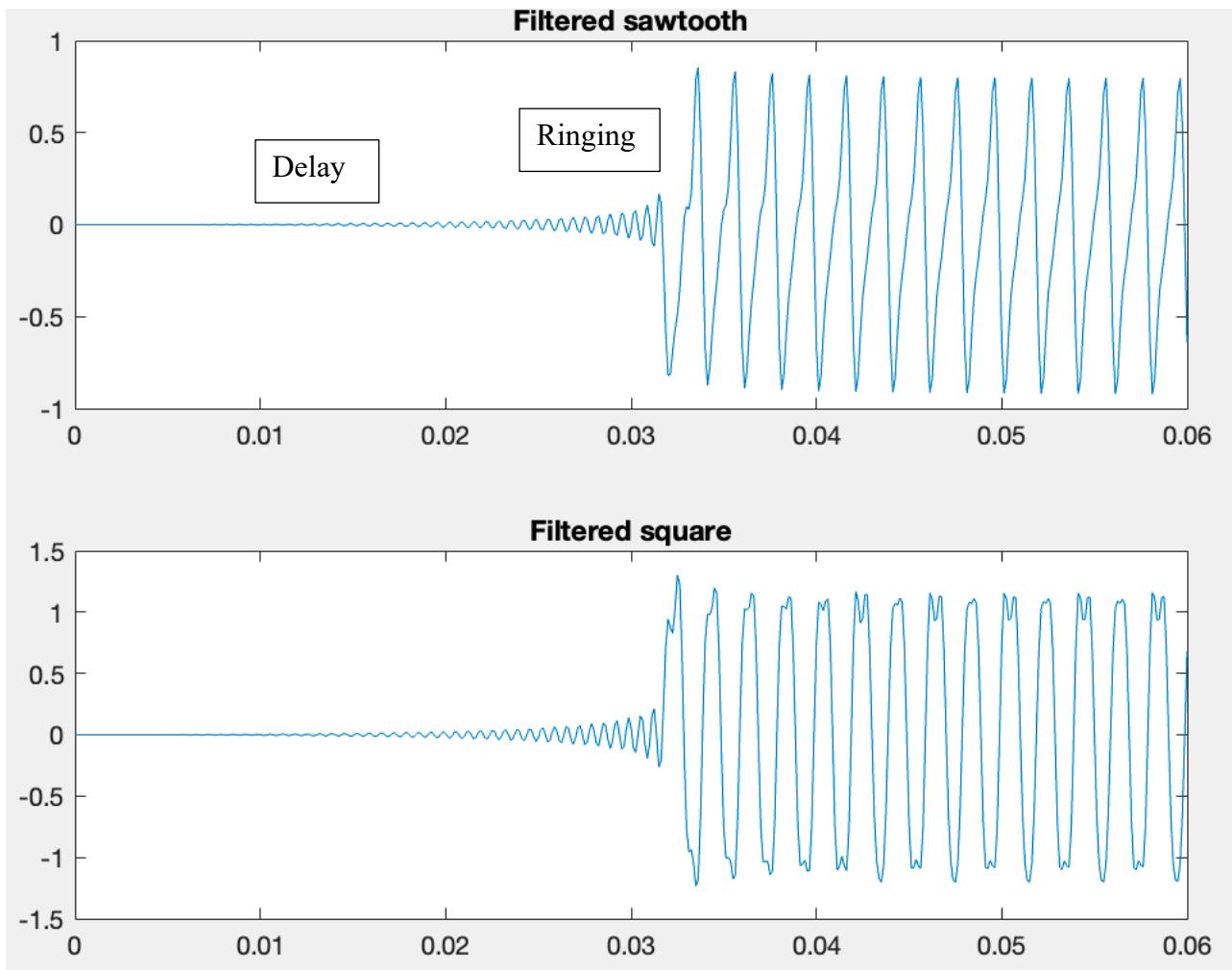
%the difference between conv and filter lies in the length of the result
%(with filter you obtain a signal with the same length of the input signal by
%throwing away the other samples,
%with conv you obtain a length that has the classical convolution length).

% Plot the results
figure()
subplot(2,1,1)
plot(t, ySawtooth);
title('Filtered sawtooth')
subplot(2,1,2)
plot(t, ySquare)
title('Filtered square')

% Comment the results, what do you expect to see in the spectrum of the
% output signals?

```





### Exercise 2 – Overlap and Add

```
%%%%%
% Overlap and add
% DAAP course
% 2024
% Mirco Pezzoli
% Exercise 2
% %%%%%%
clear
clc
close all

%% Let's verify that windows sum to a constant.
% Define an hanning window (use command hanning) with size M = 101, and hop
% size 50%
M = 101; % window length
win = hanning(M);
hopsize = (M+1)/2;

% Define a signal x of length 1000 samples
x = zeros(1000,1);
```

```
% sum the sliding windows along the signal
figure()
for i=1:10
    start = hopsize*(i-1)+1;
    stop = start + M - 1;
    x(start:stop) = x(start:stop) + win;
    plot([start:stop],win,'r')
    hold on
end

plot(x)

%% Use a switch to select different windows and overlaps within
% 1 hanning 50% overlap
% 2 bartlett 50% overlap
% 3 rectangular 50% overlap
% 4 blackman 75% overlap
% select the window type

M = 101;

n = input('Enter a number: ');

switch n
    case 1
        win = hanning(M);
        hopsize = (M+1)/2;
    case 2
        win = bartlett(M);
        hopsize = (M-1)/2;
    case 3
        M = M-1;
        win = rectwin(M);
        hopsize = (M)/2; %we have an even window
    case 4
        M = M-1;
        win = blackman(M);
        hopsize = (M)/3;
end

x = zeros(1000,1);

for i=1:10
    start = hopsize*(i-1)+1; % Start index
    stop = start + M - 1; % Stop index
    x(start:stop) = x(start:stop) + win; % Sum the window to the signal
    plot([start:stop],win,'r')
    hold on
end

plot(x)
title('COLA results');
```

### Exercise 3

```
%%%%%
% Overlap and add
% DAAP course
% 2024
% Mirco Pezzoli
% Exercise 3
% %%%%%%
%% Implement overlap and add
% Input signal: impulse train of lenght 1000
% Filter: low-pass, Nh=33 samples
% Analysis window
% Length M=33 samples
% Hop-size: 50% (M/2+1=17)
% Type: Hanning

clc
close all
clear
%% Define the signal and the filter
L = 1000;
x = zeros(1,L); % An impulse every 50 samples
x(1:50:L) = 1;
% Define the filter
Nh = 33;           % Filter length
h = fir1(Nh-1, 0.5); % Compute the filter

% OLA window definition
M = Nh;             % Window length
w = hanning(M)';    % Type of window
R = (M-1)/2+1;       % Hop size

Nfft = 2^(ceil(log2(M+Nh-1))); % length of the FFT, next power of 2 to M+Nh-1

H = fft(h, Nfft);          % Filter in frequency domain

y = zeros(1, L+Nh-1);        % Output signal

nframes = floor((L-M)/R);    % Number of frames in the signal

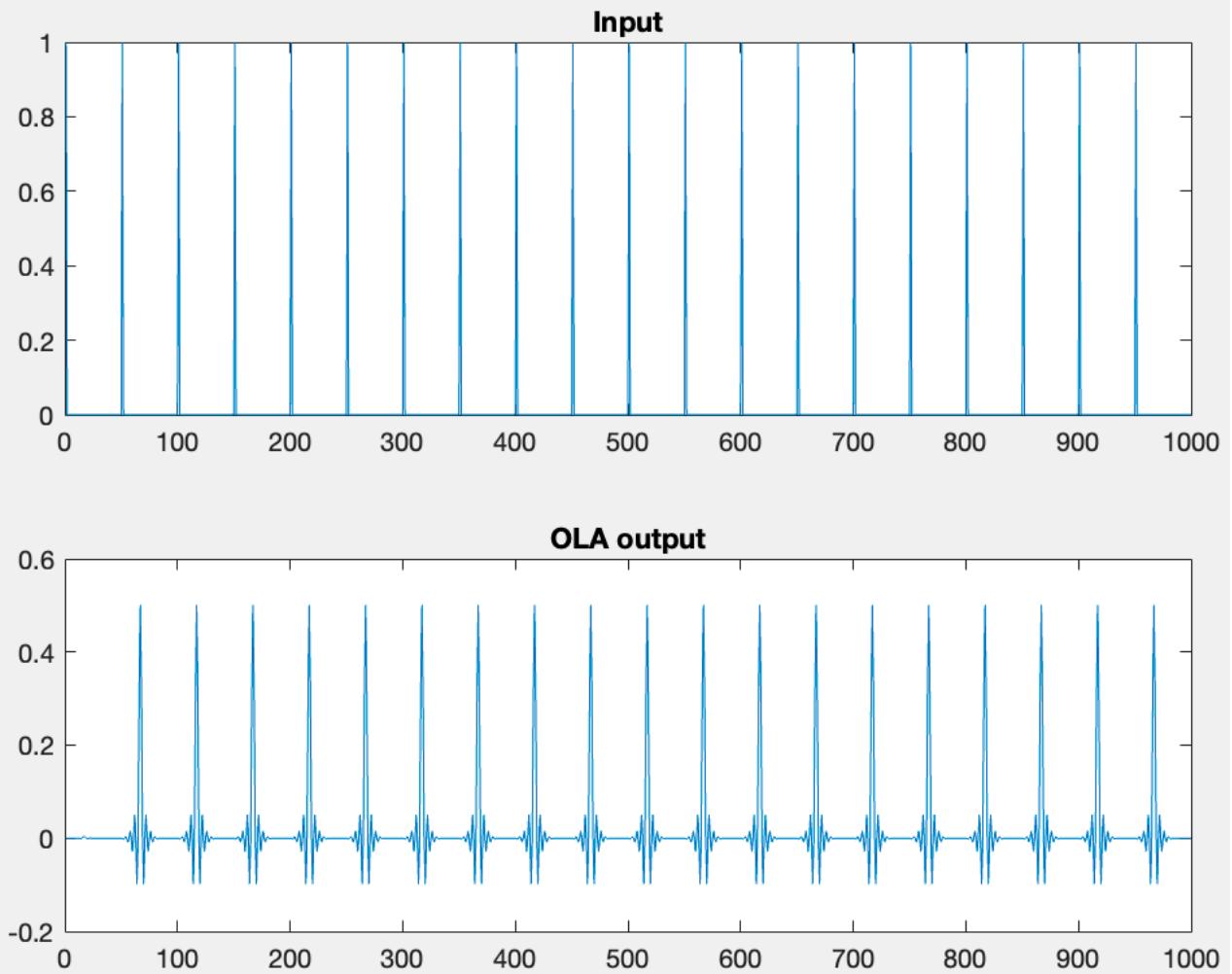
for m = 1:nframes % Sliding windows
    m_idx = (m-1)*R+1 : (m-1)*R+M;
    xm = x(m_idx) .* w;

    Xm = fft(xm, Nfft);          % Filter in the frequency domain with
    multiplication
    Ym = Xm .* H;

    ym = real(ifft(Ym));         % Return to time domain
    ola_idxs = (m-1)*R+1 : (m-1)*R + (M+Nh-1);
    ym = ym(1:M+Nh-1);
    y(ola_idxs) = y(ola_idxs) + ym; % Put everything back
end

%% Plot the input and the output of the filtering operation using subplot
figure
subplot(2,1,1)
plot(x, xlim([0, 1000])
title('Input');
```

```
subplot(2,1,2)
plot(y, xlim([0, 1000])
title('OLA output');
```

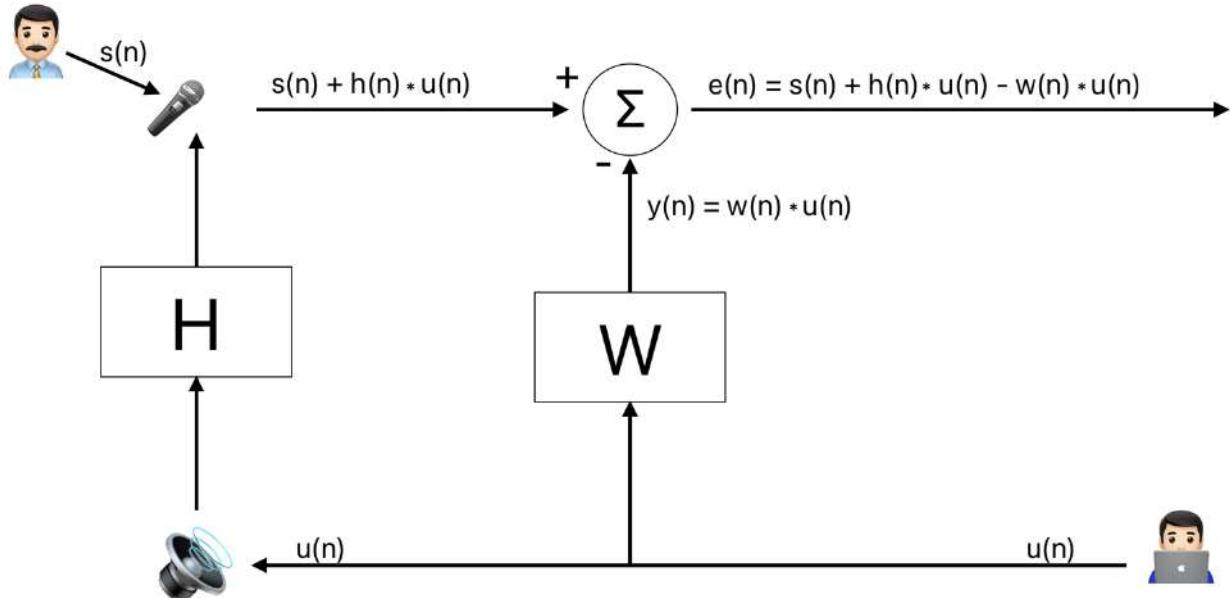


## Wiener Filtering

Today we will see how to estimate the impulse response  $h(t)$  of something. This lecture will touch topic related to statistics and analysis of stochastic signals.

### Echo cancellation

Let us suppose to be inside a room that is big enough to create echo where an online lecture is held. It can be modelled as it is done in the following scheme. There is a man 🧑 that is speaking, and its voice is captured by a microphone. There is also a loudspeaker to hear what people from home are saying. The sound from the loudspeaker is modified by the room (whose frequency response is  $H(f)$ ) and then it comes back into the microphone with the reverberation created by the room. Substantially we create a loop. But we do not want to “return back home” the sound that comes from people from home. So we try to break the loop discarding the signal  $u(n)$ . To do that we create a filter  $W(f)$  that allows us to remove information related to  $u(n)$  from the signal.



If we are able to design a good filter  $W(f)$ ,  $e(n)$  becomes exactly  $s(n)$ .

We can imagine that this condition is verified when  $w(n) = h(n)$ . In fact, we have

$$\begin{aligned} e(n) &= d(n) - y(n) = s(n) + h(n) * u(n) - w(n) * u(n) = \\ &= s(n) + h(n) * u(n) - h(n) * u(n) = s(n) \end{aligned}$$

How can we achieve this result? By relying on a technique called Wiener filtering that produces an estimate of a target signal by linear time-invariant (LTI) filtering of an observed noisy process.

Let us see it together.

### Setting

We consider a complex stationary (we remind that this means that its mean and variance does not change in time) random process with zero mean.

$$u(n), d(n), y(n), e(n) \in \mathbb{C}$$

Then we assume that our filter  $W(z)$  is a complex-valued infinite impulse response filter

$$W(z) \in \mathbb{C} \parallel \mathbb{R}$$

This means that all the coefficients that define it are complex ( $w(0), w(1), w(2), \dots \in \mathbb{C}$ )

So we know that its response can be computed through linear convolution

$$y(n) = \sum_{k=0}^{\infty} w_k^* u(n-k)$$

To identify how good our system is, we need a cost function: this function is the MSE that we call

$$J = E\{|e^2(n)|\} = E\{e(n)e^*(n)\}$$

where  $e(n) = d(n) - y(n)$  is the error.

Our desire is  $y(n)$  to match  $d(n)$ . At the end, we have an optimisation problem, so we look for the optimum filter.

The optimum filter is the one that minimises the Cost Function  $J$  so

$$\hat{W}(z) = \arg \min_{W(z)} J$$

We remind that the **arguments of the minima** (argmin) of a function are the input points at which a function output value is minimised. So, in our case, the coefficient of  $W(z)$  for which  $J$  is minimised.

### Solving

How can we solve an optimisation problem? We evaluate the derivative of  $J$  and we put it equal to zero. But, to do so we have to previously look at how  $J$  is defined.

Since the filter  $w(k)$  is complex, it has a real and an imaginary part:

$$w(k) = a(k) + jb(k)$$

We derive it. Since it is complex we have to use nabla, so we look for all the partial derivative of  $w$ .

$$\nabla k = \frac{\partial}{\partial a_k} + j \frac{\partial}{\partial b_k}$$

So we compute the gradient of the cost function and we put it equals to zero.

$$\nabla_k J = \frac{\partial J}{\partial a_k} + j \frac{\partial J}{\partial b_k} = 0$$

Now, we have our target, and we have nothing to do but compute it.

Let us start by expanding  $j$  definition.

$$\begin{aligned} \nabla_k J &= \frac{\partial E\{e(n)e^*(n)\}}{\partial a_k} + j \frac{\partial E\{e(n)e^*(n)\}}{\partial b_k} \\ &\stackrel{\text{Linearity}}{=} \\ &= E \left\{ \frac{\partial e(n)}{\partial a_k} e^*(n) + \frac{\partial e^*(n)}{\partial a_k} e(n) + j \frac{\partial e(n)}{\partial b_k} e^*(n) + j \frac{\partial e^*(n)}{\partial b_k} e(n) \right\} = \end{aligned}$$

Let us evaluate the derivatives:

- $\frac{\partial e(n)}{\partial a_k} e^*(n) \underset{e(n)=d(n)-y(n)}{\equiv} \frac{\partial[d(n)-y(n)]}{\partial a_k} e^*(n) \underset{y(n)=w(n)*u(n)=\sum_{l=0}^{\infty} w_l^* u(n-l)}{\equiv} \frac{\partial[d(n)-\sum_{l=0}^{\infty} w_l^* u(n-l)]}{\partial a_k} e^*(n) =$

$$\underset{w(k)=a(k)+jb(k)}{\equiv} \left[ \frac{\partial d(n)}{\partial a_k} - \frac{\partial \sum_{l=0}^{\infty} (a(l) + jb(l))^* u(n-l)}{\partial a_k} \right] e^*(n) \underset{\frac{\partial d(n)}{\partial a_k}=0}{\equiv}$$

*since  $a_k$  is the real part of  $w(k)$  and has nothing to do with  $d(n)$*

$$= - \frac{\partial \sum_{l=0}^{\infty} (a(l) + jb(l))^* u(n-l)}{\partial a_k} e^*(n) =$$

$$= - \frac{\partial \sum_{l=0}^{\infty} (a(l))^* u(n-l)}{\partial a_k} e^*(n) - \frac{\partial \sum_{l=0}^{\infty} (jb(l))^* u(n-l)}{\partial a_k} e^*(n) =$$

$$= - \frac{\partial \sum_{l=0}^{\infty} a(l) u(n-l)}{\partial a_k} e^*(n) + j \frac{\partial \sum_{l=0}^{\infty} b(l) u(n-l)}{\partial a_k} e^*(n) \underset{\text{The second term is } 0 \text{ since } b \text{ is the imaginary part and there is no reason why it has to depends to the real part}}{\equiv}$$

*$l$  goes from zero to  $\infty$  but the only term that is not deleted by the derivative (so the only one that depends on  $a_k$ ) is the one for which  $k=l$*

$$= - \frac{\partial \sum_{l=0}^{\infty} a(l) u(n-l)}{\partial a_k} e^*(n) \underset{\text{from zero to } \infty \text{ but the only term that is not deleted by the derivative (so the only one that depends on } a_k \text{) is the one for which } k=l}{\equiv} \left[ - \dots - \frac{\partial [a(k) u(n-k)]}{\partial a_k} \right] e^*(n) =$$

$$= -u(n-k) e^*(n)$$

- $\frac{\partial e^*(n)}{\partial a_k} e(n) \underset{e(n)=d(n)-y(n)}{\equiv} \frac{\partial[d(n)-y(n)]^*}{\partial a_k} e(n) \underset{y(n)=w(n)*u(n)=\sum_{l=0}^{\infty} w_l^* u(n-l)}{\equiv} \frac{\partial[d(n)-\sum_{l=0}^{\infty} w_l^* u(n-l)]^*}{\partial a_k} e(n) =$

$$\underset{w(k)=a(k)+jb(k)}{\equiv} \left[ \frac{\partial d(n)}{\partial a_k} - \frac{\partial \sum_{l=0}^{\infty} (a(l)+jb(l))^* u(n-l)}{\partial a_k} \right]^* e(n) = - \left[ \frac{\partial \sum_{l=0}^{\infty} (a(l)+jb(l))^* u(n-l)}{\partial a_k} \right]^* e(n) =$$

$$= - \frac{\partial \sum_{l=0}^{\infty} a(l) u^*(n-l)}{\partial a_k} e(n) + \frac{\partial \sum_{l=0}^{\infty} jb(l) u^*(n-l)}{\partial a_k} e(n) = - \frac{\partial \sum_{l=0}^{\infty} a(l) u^*(n-l)}{\partial a_k} e(n) =$$

$$\left[ - \dots - \frac{\partial a(k) u^*(n-k)}{\partial a_k} \right] e(n) = -u^*(n-k) e(n)$$
  

- $\frac{\partial e(n)}{\partial b_k} e^*(n) \underset{e(n)=d(n)-y(n)}{\equiv} \frac{\partial[d(n)-y(n)]}{\partial b_k} e^*(n) \underset{y(n)=w(n)*u(n)=\sum_{l=0}^{\infty} w_l^* u(n-l)}{\equiv} \frac{\partial[d(n)-\sum_{l=0}^{\infty} w_l^* u(n-l)]}{\partial b_k} e^*(n) =$

$$\underset{w(k)=a(k)+jb(k)}{\equiv} \left[ \frac{\partial d(n)}{\partial b_k} - \frac{\partial \sum_{l=0}^{\infty} (a(l)+jb(l))^* u(n-l)}{\partial b_k} \right] e^*(n) = - \frac{\partial \sum_{l=0}^{\infty} (a(l)+jb(l))^* u(n-l)}{\partial b_k} e^*(n) =$$

$$= - \frac{\partial \sum_{l=0}^{\infty} a(l) u(n-l)}{\partial b_k} e^*(n) + j \frac{\partial \sum_{l=0}^{\infty} b(l) u(n-l)}{\partial b_k} e^*(n) = +j \frac{\partial \sum_{l=0}^{\infty} b(l) u(n-l)}{\partial b_k} e^*(n) =$$

$$\left[ \dots + j \frac{\partial b(k) u(n-k)}{\partial b_k} \right] e^*(n) = ju(n-k) e^*(n)$$

$$\begin{aligned}
\bullet \quad & \frac{\partial e^*(n)}{\partial b_k} e(n) \underset{e(n)=d(n)-y(n)}{=} \frac{\partial [d(n)-y(n)]^*}{\partial b_k} e(n) \underset{y(n)=w(n)*u(n)=\sum_{l=0}^{\infty} w_l^* u(n-l)}{=} \frac{\partial [d(n)-\sum_{l=0}^{\infty} w_l^* u(n-l)]^*}{\partial b_k} e(n) = \\
& \underset{w(k)=a(k)+jb(k)}{=} \left[ \frac{\partial d(n)}{\partial b_k} - \frac{\partial \sum_{l=0}^{\infty} (a(l)+jb(l))^* u(n-l)}{\partial b_k} \right]^* e(n) = - \left[ \frac{\partial \sum_{l=0}^{\infty} (a(l)+jb(l))^* u(n-l)}{\partial b_k} \right]^* e(n) = \\
& = - \frac{\partial \sum_{l=0}^{\infty} a(l) u^*(n-l)}{\partial b_k} e(n) + \frac{\partial \sum_{l=0}^{\infty} jb(l) u^*(n-l)}{\partial b_k} e(n) = j \frac{\partial \sum_{l=0}^{\infty} b(l) u^*(n-l)}{\partial b_k} e(n) = \\
& \left[ \dots + \frac{\partial b(k) u^*(n-k)}{\partial a_k} \right] e(n) = -ju^*(n-k)e(n)
\end{aligned}$$

So, we obtain:

$$\nabla_k J = E\{-u(n-k)e^*(n) - u^*(n-k)e(n) - u(n-k)e^*(n) + u^*(n-k)e(n)\}$$

We group real and imaginary part obtaining:

$$\nabla_k J = -2E\{u(n-k)e^*(n)\}$$

We are now ready to find the minimum.

In order to find the solution, we start from the assumption that we already know what we want to achieve. In fact, we want to find that

$$\nabla_k J = 0 \quad \forall k$$

And we imagine that this happens for the following coefficients:  $w_o(1), w_o(2), \dots$

If we know the optimal coefficients, we can also know the optimal error that we call  $e_o(n)$ .

Let focus better on what are we doing: we propose a solution expressed by a series of coefficient to see which is the relation between those coefficients and the expression. By finding it, we will be able to explicate them.

Since we know which is the optimal result, we just have to substitute it in the  $\nabla_k J$ .

So,

$$\nabla_k J = -2E\{u(n-k)e_o^*(n)\} = 0, \quad k = 0, 1, 2, \dots$$

But hey! We notice that what we write is the orthogonality condition ( $e_o(n) \perp u(n)$ ) because we are evaluating the correlation between  $u(n)$  and  $e_o(n)$  and we are asking that it was equal to 0. This means that if the  $u(n)$  and  $e_o(n)$  are completely decorrelated, there is no information in  $e_o(n)$ , the output, related to  $u(n)$ , the looped signal. We remind that we are talking in statistical terms: this means that the output process (random signal) does not tend to change in function of the looped process  $u(n)$ .

This condition is necessary and sufficient to obtain the optimal coefficient of the linear filter.

To better understand this result we go back to the definition of the error:

$$e_o = d - y = d - \sum_{k=0}^{\infty} w_k^* u(n-k) \underset{\substack{\text{we have removed} \\ \text{all the information} \\ \text{related to } u}}{=} d$$

From a corollary of the orthogonality property, we know that

$$E\{y_0(n)e_o(n)\} = 0$$

Where  $y_0(n)$  is the optimal output that is the process we have as output from the filter  $W(f)$  when the cost function is minimised.

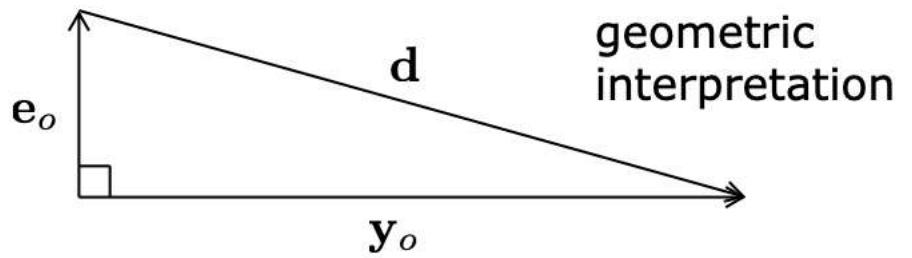
Why?

$$E\{y_0(n)e_o(n)\} = E\left\{\sum_{k=0}^{\infty} w_o^* u(n-k) e_o(n)\right\}$$

Since  $w_o(n)$  is the  
 impulse response  
 of a filter, it is  
 not a statistical  
 processes. Thus it  
 is "transparent" to  $E\{\cdot\}$

$$\sum_{k=0}^{\infty} w_o^* \underbrace{E\{u(n-k)e_o(n)\}}_{=0} = 0$$

Let us see how these vectors are related with each other: when the filter is its optimum conditions,  $y_0$  and  $e_o$  are orthogonal. So, the lower is the error, the more matched are  $y_0$  and  $d$  as we can see from the geometric diagram on the right.



There are several coefficients for which we get closer to this result, it might be useful to find a way to compare the performance of different filters. We do not use the MSE because it depends on the energy of the signal. So we define a normalised MSE to have a parameter that does not depend in any way from the signal put as input.

Some definitions:

- $\hat{d} \triangleq y_o$  Filter output at the optimal coefficients
- $J_{min} \triangleq E\{|e_o(n)|^2\}$  Minimum value of our cost function
- $\sigma_d^2 = E\{|d(n)|^2\}$  Variance of the desired signal  $d$
- $\sigma_{\hat{d}}^2 = E\{|\hat{d}(n)|^2\}$  Variance of the desired signal  $\hat{d}$  when the filter is its optimum conditions

The optimum is defined as

$$e_o(n) = d(n) - \hat{d}(n) \rightarrow d(n) = e_o(n) + \hat{d}(n)$$

We can also derive, by applying the corollary of the orthogonality principle (so without considering the term of the mutual correlation between  $d$  and  $\hat{d}$ , that

$$\sigma_d^2 = J_{min} + \sigma_{\hat{d}}^2 \rightarrow J_{min} = \sigma_d^2 - \sigma_{\hat{d}}^2$$

Hence, we are ready to introduce the normalised MSE

$$\varepsilon \triangleq \frac{J_{min}}{\sigma_d^2}$$

Since it is normalised, it is independent to the fact that the signal is strong or weak.  
From that expression we can derive that

$$\varepsilon \triangleq \frac{J_{\min}}{\sigma_d^2} \underset{J_{\min} = \sigma_d^2 - \sigma_{\hat{d}}^2}{=} 1 - \frac{\sigma_{\hat{d}}^2}{\sigma_d^2}$$

This assures us that  $0 \leq \varepsilon \leq 1$

It is  $= 1$  if the filter is completely bad done, since  $\sigma_{\hat{d}}^2 = 0$ .

It is  $= 0$  if the filter is perfect, since  $\sigma_{\hat{d}}^2 = \sigma_d^2$ . This means that there is a complete agreement between  $\hat{d}(n)$  and  $d(n)$ .

Let us compute the optimal coefficients by simply substitute them in the definition of the cost function

$$E\{u(n-k)e_o^*(n)\} = 0 \quad \forall k = 0, \dots, \infty$$

$$E\left\{u(n-k) \left[ d^*(n) - \sum_{l=0}^{\infty} w_0(l)u^*(n-l) \right] \right\} = 0 \quad \forall k = 0, \dots, \infty$$

$$E\left\{u(n-k)d^*(n) - \sum_{l=0}^{\infty} u(n-k)w_0(l)u^*(n-l)\right\} = 0 \quad \forall k = 0, \dots, \infty$$

$$E\{u(n-k)d^*(n)\} - w_0(l)E\left\{\sum_{l=0}^{\infty} u(n-k)u^*(n-l)\right\} = 0 \quad \forall k = 0, \dots, \infty$$

$$\sum_{l=0}^{\infty} w_0(l) \underbrace{E\{u(n-k)u^*(n-l)\}}_{r(l-k)} = \underbrace{E\{u(n-k)d^*(n)\}}_{p(-k)} \quad \forall k = 0, \dots, \infty$$

*Autocorrelation of the input*                   *cross-correlation of the input and the desired response*

Informal meaning of these terms:

- $r(l-k)$ : the autocorrelation of the input expresses similar is the statistical observation of a random variable extracted from the input process at an instant  $n_0 - k$  and another one extracted from the same process at an instant  $n_0 - l$ .
- $p(-k)$ : the cross-correlation of the input and the desired response shows us how similarly varies a variable extracted from the input process at an instant  $n_0$  and another one extracted from the desired response at an instant  $n_0 - k$

$$\sum_{l=0}^{\infty} w_0(l)r(l-k) = p(-k) \quad \forall k = 0, \dots, \infty$$

This is nothing but the Wiener-Hopf equation since it represents is a linear system of infinite equations, which defines the optimum filter coefficients. Given the fact that we deal with an infinite number of equations, we will have a solution (a coefficient) for one of them so an infinite number of coefficients.

In practice, this is not possible because with an infinite number of coefficients we would obtain an IIR filter that can be unstable. We prefer working with FIR instead, as they are always stable. For this reason, we truncate the response by considering just  $M$  coefficients.

$$\sum_{l=0}^{\infty} w_0(l)r(l-k) = p(-k) \quad \forall k = 0, \dots, M-1$$

So, by considering all the values of all the instants of time, we will have the input vector, (the input to the Wiener-Hopf filter, not of the entire system)

$$\underline{u}(n) = [u(n), u(n-1), \dots, u(n-M+1)]^T$$

From which we obtain the cross-correlation vector

$$\underline{p}(k) = E\{\underline{u}(n)\hat{d}^*(n)\} = [p(0), p(-1), \dots, p(-M+1)]^T$$

Where  $\underline{d}(n)$  is the desired response vector from the Wiener-Hopf filter when it is defined with the following optimum coefficients

$$\underline{w}_0 = [w_0(0), w_0(1), \dots, w_0(M-1)]^T$$

We can represent the autocorrelation in a matrix fashion obtaining the so-called auto correlation matrix

$$\underline{\underline{R}} = E\{\underline{u}(n)\underline{u}^*(n)\} = \begin{bmatrix} r(0) & r(1) & \cdots & r(M-1) \\ r^*(1) & r(0) & \cdots & r(M-2) \\ \vdots & \vdots & \ddots & \vdots \\ r^*(M-1) & r^*(M-2) & \cdots & r(0) \end{bmatrix}$$

From which we can write

$$\underline{\underline{R}} \underline{w}_0 = \underline{p}$$

We can find the optimum coefficients by inverting the relation:

$$\underline{w}_0 = \underline{\underline{R}}^{-1} \underline{p}$$

### Graphical interpretation

There is also another way to derive the Wiener-Hopf equation that consists of examining the dependence of the cost function  $J$  on the tap weights of the filter.

So, we start by looking at the definition of the cross function (the MSE of the variance of  $e$ )

$$J = E\{e(n)e^*(n)\} \stackrel{\substack{= \\ \text{FIR model} \\ \text{we go from 0 to } M-1}}{}$$

$$\begin{aligned}
&= E \left\{ \left[ d(n) - \sum_{k=0}^{M-1} w^*(k)u(n-k) \right] \left[ d^*(n) - \sum_{k=0}^{M-1} w(k)u^*(n-k) \right] \right\} = \\
&= \underbrace{E\{d^2(n)\}}_{\sigma_d^2} - \sum_{k=0}^{M-1} w^*(k) \underbrace{E\{u(n-k)d^*(n)\}}_{p(-k)} - \sum_{k=0}^{M-1} w(k) \underbrace{E\{u^*(n-k)d(n)\}}_{p^*(-k)} + \sum_{k=0}^{M-1} \sum_{l=0}^{M-1} w^*(k)w(l) \underbrace{E\{u(n-k)u(n-l)\}}_{r(l-k)} = \\
&= \sigma_d^2 - \sum_{k=0}^{M-1} w^*(k)p(-k) - \sum_{k=0}^{M-1} w(k)p^*(-k) + \sum_{k=0}^{M-1} \sum_{l=0}^{M-1} w^*(k)w(l)r(l-k)
\end{aligned}$$

What we obtained is a kind of parabola in M dimension. In fact, we have a constant term  $\sigma_d^2$ , a term that varies linearly with  $w(k)$  or  $w^*(k)$  and a last term that varies linearly with the product between  $w^*(k)$  and  $w(l)$ . Formally, it is not a parabolic shape but, since the cost function is defined over a domain of  $M$  samples, it is a bowl-shaped ( $M+1$ )-dimensional surface ( $M$  for the samples and 1 for the codomain) with  $M$  degrees of freedom.

We remind the degree of freedom represents the number of independent parameters of the system that we need to know to determinate the condition of the system itself.

This type of function is called **error performance surface**. As it happens for every “parabolic” shape, we know that it will exhibit a unique minimum.

So, let us evaluate the derivative in order to find it

$$\nabla_k J = \frac{\partial J}{\partial a_k} + j \frac{\partial J}{\partial b_k} = -2p(-k) + 2 \sum_{l=0}^{M-1} w(l)r(l-k) = 0$$

And we are back to the Wiener-Hopf equation. We just have to find the value of this minimum! We start from the convolution expressed in vectorial fashion (we remind that the vectorial product is the sum of the products as the convolution is)

$$\hat{d} = \sum_{k=0}^{M-1} w_o^*(k)u(n-k) = \underline{w}_o^H \underline{u}$$

And we express the variance in function of these vectors:

$$\sigma_{\hat{d}}^2 = E\{|\hat{d}^2|\} = E\{\hat{d}\hat{d}^*\} = E\{\underline{w}_o^H \underline{u} \underline{u}^H \underline{w}_o\} = \underline{w}_o^H \underline{R} \underline{w}_o \stackrel{\underline{R} \underline{w}_o = \underline{p}}{=} \underline{w}_o^H \underline{p} = \underline{p}^H \underline{w}_o \stackrel{\underline{w}_o = \underline{R}^{-1}\underline{p}}{=} \underline{p}^H \underline{R}^{-1} \underline{p}$$

We plug it into the definition of  $J_{\min}$

$$J_{\min} = \sigma_d^2 - \sigma_{\hat{d}}^2 = \sigma_d^2 - \underline{p}^H \underline{R}^{-1} \underline{p}$$

But where are the coefficient values? We are just expressed  $J_{\min}$  in function of the auto correlation matrix and of the cross-correlation vector. We now have to retrieve the coefficients from this expression rewriting the error-performance surface in matrix form:

$$\begin{aligned}
J(\underline{w}) &= \sigma_d^2 - \sum_{k=0}^{M-1} w^*(k)p(-k) - \sum_{k=0}^{M-1} w(k)p^*(-k) + \sum_{k=0}^{M-1} \sum_{l=0}^{M-1} w^*(k)w(l)r(l-k) \\
&\quad \stackrel{\text{from convolution to matrix fashion}}{=} \\
&= \sigma_d^2 - \underline{w}^H \underline{p} - \underline{p}^H \underline{w} + \underline{w}^H \underline{R} \underline{w}
\end{aligned}$$

We rearrange the terms, and we use a trick in order to make it: we add and subtract different definitions of  $w$

$$\begin{aligned} J(\underline{w}) &= \sigma_d^2 - \underline{w}^H \underline{p} - \underline{p}^H \underline{w} + \underline{w}^H \underline{\underline{R}} \underline{w} + \underbrace{\underline{w}_o^H \underline{\underline{R}} \underline{w}_o - \underline{w}_o^H \underline{\underline{R}} \underline{w}_o}_{\text{Trick}} \stackrel{\underline{p} = \underline{\underline{R}} \underline{w}_o}{=} \\ &= \sigma_d^2 - \underline{w}^H \underline{\underline{R}} \underline{w}_o - (\underline{\underline{R}} \underline{w}_o)^H \underline{w} + \underline{w}^H \underline{\underline{R}} \underline{w} + \underline{w}_o^H \underline{\underline{R}} \underline{w}_o - \underline{w}_o^H \underline{\underline{R}} \underline{w}_o = \end{aligned}$$

Reminder: associative product of Hermitian matrices

$$(AB)^H = [(AB)^T]^{-1} = (B^T A^T)^{-1} = (B^T)^{-1} (A^T)^{-1} = B^H A^H$$

Thus:

$$(\underline{\underline{R}} \underline{w}_o)^H = \left[ (\underline{\underline{R}} \underline{w}_o)^T \right]^{-1} = (\underline{w}_o^T \underline{\underline{R}}^T)^{-1} = (\underline{w}_o^T)^{-1} (\underline{\underline{R}}^T)^{-1} = \underline{w}_o^H \underline{\underline{R}}^H \stackrel{\underline{R}^H = (\underline{\underline{R}}^T)^{-1}}{=} \underline{w}_o^H \underline{\underline{R}}$$

$$\begin{aligned} &= \sigma_d^2 - \underline{w}^H \underline{\underline{R}} \underline{w}_o - \underline{w}_o^H \underline{\underline{R}} \underline{w} + \underline{w}^H \underline{\underline{R}} \underline{w} + \underline{w}_o^H \underline{\underline{R}} \underline{w}_o - \underline{w}_o^H \underline{\underline{R}} \underline{w}_o = \\ &= \underbrace{\sigma_d^2 - \underline{w}_o^H \underline{\underline{R}} \underline{w}_o}_{J_{\min}} - \underline{w}^H \underline{\underline{R}} \underline{w}_o - \underline{w}_o^H \underline{\underline{R}} \underline{w} + \underline{w}^H \underline{\underline{R}} \underline{w} + \underline{w}_o^H \underline{\underline{R}} \underline{w}_o = \\ &= J_{\min} - \underline{w}^H \underline{\underline{R}} \underline{w}_o - \underline{w}_o^H \underline{\underline{R}} \underline{w} + \underline{w}^H \underline{\underline{R}} \underline{w} + \underline{w}_o^H \underline{\underline{R}} \underline{w}_o = \\ &= J_{\min} - (\underline{w}^H - \underline{w}_o^H) \underline{\underline{R}} \underline{w}_o + (\underline{w}^H - \underline{w}_o^H) \underline{\underline{R}} \underline{w} = \\ &= J_{\min} - (\underline{w} - \underline{w}_o)^H \underline{\underline{R}} \underline{w}_o + (\underline{w} - \underline{w}_o)^H \underline{\underline{R}} \underline{w} = \\ &= J_{\min} + (\underline{w} - \underline{w}_o)^H \underline{\underline{R}} (\underline{w} - \underline{w}_o) \end{aligned}$$

From this expression is obvious that when  $\underline{w} = \underline{w}_o$ , the cost function reaches its minimum. We re-write it in the so-called canonical form by diagonalising the matrix  $\underline{w}$  and make its element uncorrelated. Everything here is based on the eigenvalue decomposition of  $\underline{R}$ . In fact, we know that can write  $\underline{R}$  as the product of three matrixes:

$$\underline{\underline{R}} = \underline{\underline{Q}} \underline{\Lambda} \underline{\underline{Q}}^H$$

Where:

$$\underline{\underline{Q}} \underline{\underline{Q}}^H = I \quad \underline{\Lambda} = \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_M \end{pmatrix}$$

So, we change the basis:

$$\underline{\underline{v}} = \underline{\underline{Q}}^H (\underline{w} - \underline{w}_o) = [v_1, v_2, \dots, v_M]^T$$

Expressed in the canonical form the cost function will be equal to

$$J = J_{\min} + \underline{v}^H \underline{\Lambda} \underline{v} = J_{\min} + \sum_{k=1}^M \lambda_k v_k v_k^* = J_{\min} + \sum_{k=1}^M \lambda_k |v_k|^2$$

So, simply speaking, we have changed the basis of our cost function from  $w$  to the basis that maximise the uncorrelation in  $R$ , making in this way  $R$  diagonal.

## Audio Restoration

First of all, we have a huge number of contents generated. High quality contents have increased, and with them our threshold of quality. This creates a problem with everything that has been created in the past (the recorded created during the last 150 years): we need to restore it to perceive it as a good product.

Even if we just ignore everything that is before the era of recording, there is an enormous amount of music that needs to be processed. The issue of recuperating and restore has been always there. Even though we like to hear new music, we have to remind that there is a value in recording that cannot be discarded. The reason why science progressing exponentially is because of documentation. We stand on the shoulders of other giants, and music has to do the same thing: we have to stand on the shoulder of other giants.

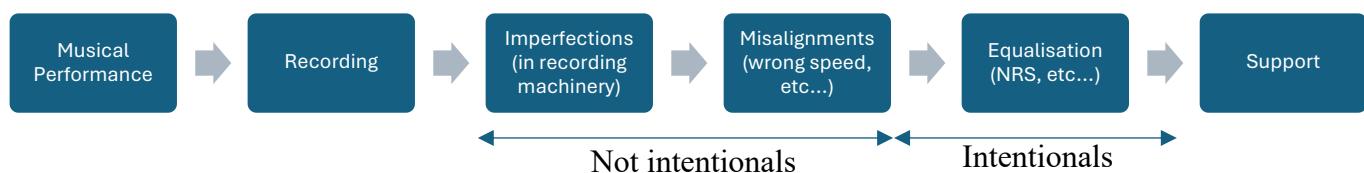
So we need to restore for three reasons:

1. Not to lose the recordings.
2. To keep up with the perception of quality.
3. To be able to change the sound in order to create, in retrospect, a new microphone setting. This happens because microphoning, choosing the position of the microphones, is a form of art and the perception of a real sound changes in human as time goes by.

The point is: you are never done! Multimedia has to do with perception so, since the perception changes in time and people want to experience new things, we will have always something to work on.

Let us see which are the differences between the process of recording and restoring

### Degradation



We start from the phase of the musical performance, then we record the performance itself. Sometimes, we do not know how the recording is made therefore we cannot exploit information coming from the position of microphones and so on. Anyway, every recording comes with some kind of imperfections: firstly, the one related to the machinery (hissing noise, clicks and so on), then the problems related to misalignments. At the end we have some intentional degradations. It could be strange but, in the past, there were some standards in which the equalisation process degrades the signal. This was done, generally, for competitive reasons. At the end we have supports. Since most of them are physical and analogical objects, they are strongly subjected to the wear of time.

### Restoring

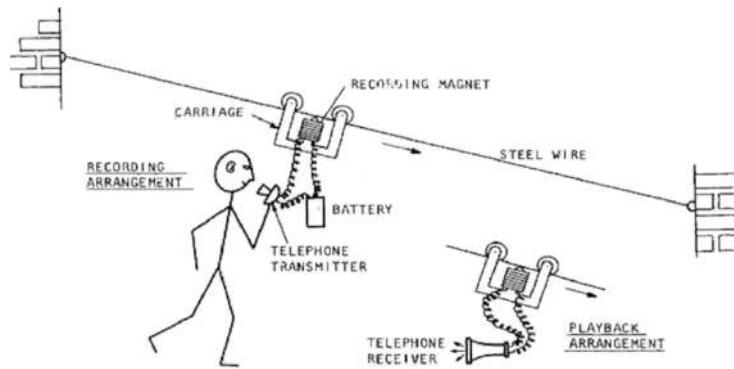


Nowadays, we are trying to digitalise contents as much as possible. In order to do that, we have priorly to restore the audio signals. This is generally done to remaster old tracks with new and more accurate techniques. To go back to the original track, we have nothing to do but following the inverse path.

## Tape recording - A little bit of history

### 1899: Paulsen's steel wire recording

The drawing on the right is a system that describes the very first version of magnetic recording. It was invented in 1889 by the Danish-American engineer Valdemar Poulsen. You simply walk moving a carriage that leaves magnetic residuals on a tape placed on a steel wire stretched between two walls. In order to reproduce the recorded sound, the carriage has to be connected to a loudspeaker. Wire recorders had poor audio quality and were replaced by magnetic tape recording in the 1940s.



### 1935: AEG's first tape recorder (magnetophone)

The «Magnetophon» was the first reel-to-reel tape recorder developed by the German electronics company AEG in the 1930s, based on the magnetic tape invention by Fritz Pfleumer. Later models introduced the concept of AC tape bias, which improved the sound quality by largely eliminating background hiss with enormous quality improvement. Adolf Hitler used these machines to perform what appeared to be live broadcasts from one city while he was in another.

Two later model Magnetophones were taken to the US after the war, which included both the newer oxide coated PVC tape developed by I.G. Farben (BASF division) as well as the AC bias system.



The Army officer who tracked them down, Jack Mullin, would use these machines as the basis of his own designs, which he first demonstrated to the San Francisco chapter of the Institute of Radio Engineers in May 1946, and later at the MGM Studios in Hollywood in October of that year. Attending the SF demo were Ampex engineers who were inspired to design their own reel-to-reel recorder based on Mullin's modified Magnetophon. A demonstration of the Ampex Model 200A was arranged with Bing Crosby. The playback audio quality was good enough to get Crosby to agree to work with them (by arranging financial support for start-up manufacturing). The Ampex 200A went into production and within three years most major recording studios had purchased one.



### 1936: TEFIFON

The TEFIFON, manufactured and sold in Germany, could be seen as the grandparent of the cassette tape. It was, in fact, a vinyl belt housed in a cassette. The cassette was loaded with an endlessly looped reel (similar to the 8-track tape) of plastic tape. Depending on the size of the tape, the play time could range from 15 minutes to 4 hours. Tefifon production at its main plant in Porz am Rhein ceased in 1965.



### 1951 - MINIFON - Analog, Magnetic wire

Due to its small size, it was adopted by many governments as a “spy” recorder. The Minifon was a battery-operated miniature tape recorder developed by German electrical engineer Willi Draheim. The wire would run at a speed of 30 cm/s or about 11.8 ips.



### 1953 - PYE AND TELEFUNKEN MAGNETIC DISCS

These discs allowed users to record on a magnetic disc instead of tape. The recording was made on a flexible plastic, pre-grooved disc which was coated with magnetic compound. The disc had a running time of 10 minutes. The recording could be played back through a loudspeaker similar to the standard Gramophone records of the time. These were manufactured well into the 1960's.



### 1954 - GRUNDIG STENO RETTE

This was used in the Stenorette office dictation machine and was one of the first to use magnetic tape. Similar to an 8-Track tape, this had a single reel. The tape was 1/4" wide and normally had a running time of approximately 25-30 minutes. These were manufactured into the 1970's.



### 1955 - MOHAWK MIDGETAPE

The Midgetape was an early version of a portable tape recorder. It was manufactured fairly well with a sturdy metal casing. The 1/4" tape allowed for roughly 45 minutes of recording per side. These were manufactured into the mid 1960's.



### 1957 - DICTET

Developed by the Dictaphone Corporation in 1957. It was used in the very first dictation machine that used magnetic tape cassettes. This cassette used 1/4" tape and ran at 2.48 inches per second.



### 1958 - RCA SOUND TAPE CARTRIDGE

Developed in 1958. Also known as a Magazine Loading Cartridge or Sound Tape. Like a reel to reel, it is a 1/4" magnetic tape that ran at standard speed of 3.75 inches per second (9.525 cm/s) and was designed to offer reel to reel tape recording quality in a convenient format. The same design was introduced by Phillips in 1963 on the standard compact cassette. This format ultimately failed for two reasons: slow to produce machines for the home market; slow to license prerecorded music tapes. This format ceased production by 1964.



### 1959 - PROTONA MINIFON

The Minifon Attaché was a miniature tape recorder using a tape cartridge. Developed by Protona GmbH in Hamburg, Germany around 1959. The Attaché was the first all-transistor recorder developed Minifon and contained a 4-stage amplifier on a printed circuit board. The Minifon Attaché was in production until the end of the Minifon product line in 1967.



### 1959 - NAB CART TAPE

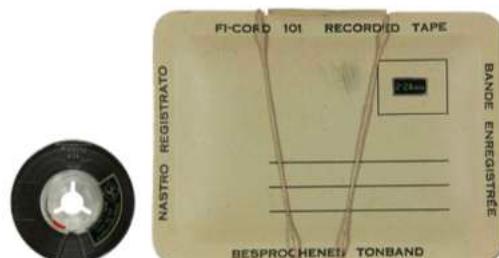
Similar to an 8-track tape. This is a 4-track radio station cart, also known as Fidelipac. NAB cart tapes were introduced in 1959 by Collins Radio. This particular cart was used by WBNS radio station in Columbus, OH in the 1980's. This was a working cart at the radio station prior to digital file formats being implemented.

Each cart had one song and had a number shown at the end of each one. To play a particular song, all you had to know is the correct cart number so that it could be pulled for air play. Despite looking like an 8-track tape, it does not play in an 8-track player without some modifications. The magnetic tape itself could be transferred into an old 8-track case and be made to play on an 8-track machine. The main difference in 4-track cartridge design from 8-tracks is that 4-tracks lack a built-in pinch roller which would grip and help move the tape; a hole is left in the cartridge for a pinch roller to be inserted from inside the 4-track player itself.



### 1960 - FI-CORD

High quality portable reel tape recorder produced in the early 60s by Fidelity Recording in Switzerland. The tape had a diameter of 2" and could record for 25 minutes. The tape ran at 7.5 or 1 7/8 ips. It was also known as the "spy recorder" of the early 1960s, since it had a detachable speaker that could be taken off the recorder for covert recording. Due to its small size, it also worked well for mailing.



### 1961 - CHANNEL MASTER 6546

Also known as the Sanyo Micro-Pack 35. Basically, a dictation machine. It was a highly superior recorder that required its own reel-to-reel monaural tape made by the company. That means that instead of the reels being side by side like a normal compact cassette tape, these are on top of each other. The tape is linked diagonally between the two reels as you can see in the picture on the right. The recorder had a speed control which give the user infinite variable speeds.



### 1961 - ORRTRONIC TAPETTE

Orrtronics was a company that made a background music system based on the original Echo-matic cartridge. Very similar to (and the predecessor of) the 8-Track tape. It was actually better-sounding but commercially unsuccessful since no record companies were interested in the format. As a result, it went nowhere and was discontinued.



### 1962 - ECHO-MATIC II

Invented by Bernard Cousino (who also invented the above Orrtronic Tapette along with John Herbert Orr). It was a two-track endless-loop magnetic tape cartridge. Anybody who loved watching the movie Big would appreciate this tape. Its most famous usage was in the Zoltan fortune teller arcade machine.



### 1963 - COMPACT CASSETTE

The Compact Cassette was introduced in 1963 and is also a magnetic tape-recording format for audio recording and playback. Like the reel to reel, cassettes had two miniature reels or spools, between which a magnetically coated plastic tape is passed and wound. The tape could be played or recorded on both sides by simply flipping the tape over. In the US, mass production of the cassette music started in 1964. Most of us will remember that during the 1980s, the cassette's popularity grew even further as a result of portable pocket recorders such as Sony's Walkman (1979). Cassettes went on to become a popular alternative to the 12-inch vinyl LP during the late 1970's. Ultimately, the sales of pre-recorded cassette tapes were overtaken by those of CDs during the early 1990s. Most music companies discontinued production of cassette tapes by late 2002 although blank cassette sales continued slightly longer.



Magnetic cassette tapes underwent several transformations over the years:

- Type I: Ferric Oxide – poor high-frequency response and excessive tape hiss
- Type II: Chromium Dioxide – better frequency resp. but not as good on low end and output levels.
- Type III: A Ferric Oxide and Chromium Dioxide mix – slightly improved low frequency response.
- Type IV: Pure metal particles – increased dynamic range and frequency response but increased head wear.



### 1964 - GRUNDIG EN3

This is a tape from the Grundig EN3 handheld voice dictation machine. The 1/4" tape could record up to 30 minutes of dictation. It's interesting to note that the microphone on this device also acted as the speaker. These were in production until the 1970's.



## 1965 - 3M CANTATA 700

The largest (9 3/8" x 9 5/8" x 1 5/8") and heaviest audio tape cartridge (a little under 3 pounds) in history. The cartridge itself holds two 8" reels with 1/4" tape. It could hold up to 700 songs (hence the name) of different styles and was mostly used in offices, restaurants and other public buildings. These were in use well into the mid 90's.



## 1965 - 8-TRACK

Popular from the mid-1960s to the late-1970s when the Cassette format took over. The tape was on a single reel, with the magnetic tape forming an endless loop that would pull from the centre of the reel, play, then wrap around the outside of the reel.



## 1966 - PLAYTAPE

Like a miniature 8-track tape. A PlayTape cartridge uses 1/8" tape and plays anywhere from 8 to 24 minutes. The introduction of home and portable players by the 4-track and 8-track manufacturers led to the ultimate demise of PlayTape.



## 1967 - NAGRA CREVETTE

Crevette (English: shrimp) was a small electromechanical tape recorder, developed in 1967 by Nagra in Switzerland especially for the French Army. The device was intended for use inside a blank submarine torpedo, and was used for recording tracking, gyro and navigational data, allowing engineers to perfect the torpedo's tracking trajectory and guidance system later.



## 1969 - MICROCASSETTE

Introduced by Olympus. The tape width of this tape is the same as that of a standard cassette, but the case is much smaller. Tape speed is 2.4 cm/s and could play 30-45 minutes, depending on the length of the tape. The tapes were mostly used for recording voice as in dictation machines and answering machines. They were also used as a medium for computer data storage and music, although to a lesser extent. The classic cassette is reported in the image just to make a comparison of the dimension.



## 1969 - MINI-CASSETTE

Introduced by Philips. The tapes could record 30 minutes and were primarily used in dictation machines. They were also employed as a data storage for the little known Philips P2000 home computer. Unlike other cassette style cartridges, the mini-cassette did not use a capstan drive system. The tape was propelled past the tape head by the reels. Although prone to wow and flutter, the lack of a capstan and a pinch roller drive meant that the tape worked well for dictation machines which repeatedly shuttled forward and backward in short distances. (less popular than microcassettes)



### 1970 - QUADRAPHONIC 8-TRACK

Also known as Quad-8 or Q8 tape. It looks almost identical to the standard 8-Track cartridges except for sensing notch near the top left of the cartridge. This allowed a machine to sense that a Quadraphonic cartridge was present. These tapes were only available until around 1978.



### 1970'S - KALAVOX AUDIO SLIDE CASSETTE

When he came across these, professor Sarti was a little surprised that he had not heard of them at some point. This is an audio slide cassette by Kalart Victor. From what he has researched, these were in production sometime during the 70's. The cartridge consisted of a 2"x2" slide holder along with a tape cartridge that had a capacity of 60 seconds where each sound slide had its own narration. A person could mount up to 40 sound slides in a special carousel tray. The tray could be placed on a normal Kodak Carousel projector in place of the traditional slide carousel.



### 1971 - STENO-CASSETTE

The Steno-Cassette was introduced by the German company Grundig in 1971. It was widely used in Germany for dictation. It is easily distinguished from other dictation cassettes due to the integrated tape counter index, showing the amount of tape available. The single sided 30-minute tape helped prevent mix-up or accidental deletion.



### 1971 - HIPAC

Successor to the PlayTape cartridge. It was a short-lived format and never gained popularity. The four audio tracks of the tape were separated into two stereo programs. Also, different from a CD, the two programs were recorded in the same direction.



### 1971 - U-MATIC

Among the first videotape cassettes. Also used later for the storage of digital audio data and audio mastering. U-matic was named after the shape of the tape path when it was threaded around the helical video head drum, which resembled the letter U. Betamax used this same type of "U-load" as well. Although this format never really had success in the household consumer market, it did have some limited success in business communication and educational television.



### 1975 - BETAMAX DIGITAL AUDIO (BETA)

## 1976 - VIDEO HOME SYSTEM (VHS)

The competition between the two formats was fierce. Betamax (Sony) was introduced in 1975 and VHS (JVC) a year later in 1976. Both formats were an analogical videocassette magnetic tape with 1/2-inch-wide magnetic tape. The wide tape allowed it to be slowly passed over the various playback and recording heads of the video cassette recorder. The Betamax had a recording time up to 2 hours while the VHS tape could record up to 4 hours. Music themed and concert setting tapes were very popular in both formats. Though Betamax was slightly superior in quality, consumers could not tell the difference, and the lower cost of the VHS recorders ultimately won.



## 1976 - ELCASET

Jointly developed by Panasonic, Sony, and Teac in 1976, but shortly lived. The name "Elcaset" may simply mean L-cassette, or large cassette, since the 1/4" tape inside is double the 1/8" width found in compact cassettes. At that time, it was felt that cassette tapes would never be capable of the same levels of performance that was available from reel-to-reel systems, but it was quite clear that cassettes were much more convenient. The Elcaset system was intended to marry the performance of reel-to-reel with cassette convenience. The cassette itself looks similar to a compact cassette, only larger—about twice the size.



## 1982 - PICOCASSETTE - Analog, Magnetic Tape

Introduced by Dictaphone in collaboration with JVC. Extremely small cassette style tape and was approximately half the size of the Microcassette. It had a tape speed of 9 mm/s and each cassette could hold up to 60 minutes of dictation. Only produced for a few years.



## Mid 1980's - BANDAI MICRO CARTRIDGE

Made by Bandai Japan. The very small endless-loop magnetic tape cartridge contained only one song and measured only 1 5/16" x 1 1/8". These were only used in a miniature replica Wurlitzer jukebox made by Leadworks, and an "Elvis in Concert" doll with built in tape player.



## 1987 - DIGITAL AUDIO TAPE (DAT)

Recording and playback medium developed by Sony. The DAT used a 3.8mm/0.15" (commonly referred to as 4mm) magnetic tape enclosed in a protective shell. The shell was approximately half the size of a standard cassette tape. DAT did not use any data compression. Because of that, the music industry strongly resisted this technology over concerns of piracy, which is why the format was never widely adopted by consumers. In 2005 Sony discontinued its remaining DAT machine models.



### 1992 - DIGITAL COMPACT CASSETTE (DCC)

Introduced by Philips and Matsushita in 1992 and marketed as the successor to the standard analogical Compact Cassette. DCC recorders could play back both standard analogical cassettes and DCC tapes. Although technically superior to the analogical cassette, the DCC's were never able to topple the cheaper cassette. DCC was discontinued in October 1996 after Philips admitted it had failed at achieving any significant market penetration.



### 1992 - NT (Cassette)

Digital memo recording system manufactured by Sony in 1992. Sometimes marketed under the name Scoopman. Extremely tiny cassettes but could hold up to 120 minutes. Manufactured until the late 1990's. NT stands for Non-Tracking, meaning the head moves at a shallower slope to that of the tracks on the tape, crossing several during each pass, albeit only reading partial data from each one. By making several passes it is possible to reassemble the complete data for each track. This considerably reduced the complexity and size of the head, and, therefore, the recorder.



### 1993 - DTRS (DIGITAL TAPE RECORDING SYSTEM)

Digital multitrack recording device introduced by the TASCAM division of the TEAC Corporation in 1993. The DA-88 was among the first affordable digital recorders available to home studios. The 8-track, 16-bit recorder captured audio to Hi-8 cassettes and was known for its reliability. It was also known as "modular digital multitrack" (MDM) machine. You could also combine multiple DA-88 devices to record 16 or more tracks. The audio was stored in the DTRS format on Hi-8mm video compact cassettes which was basically the same size as a standard compact cassette tape. It allowed up to 113 minutes of continuous recording on a single tape. As of 2012, the DTRS format is officially retired.



## Formats for studio recording

For the longest time, the system used was 16 head one. But this was the biggest one: only Pink Floyd and few more bands recorded with this format. Beatles, for example, recorded for years just on four tracks format. It is not that easy to “consolidate” (this is the term used to express the condensation of a set of 16 tracks in just two) a large number of analogical tracks. Nowadays, we record with at least 200 tracks. This was impossible in the analogical era. Let us see some formats

Inches	Type	Use
2	Open reel	Professional
1	Open reel	Professional
1/2	Cassette	Professional
1/4	Open reel	Professional
1/4	Open reel	Semi-Professional
1/4	Cassette	Stereo 8 cartridges
1/4	Cassette	Elcaset
1/4	Cassette	Compact cassette

Category	Type of tape	Inventor
IEC1	Iron oxide (normal)	BASF
IEC2	Chrome dioxide Chome-equivalents (high)	BASF
IEC3	Double layer Chrome Iron	SONY
IEC4	Metal	TDK

Tape recording uses a very specific type of speed: 76 cm/s, later reduced to fractions of it (38, 19, 9.5, 4.75 cm/s). 4.75 is the speed of a compact cassette. The higher the speed, the larger is the dynamic range. The tape size is also standardised: 1/4 of an inch (6.3 mm) for one or two tracks. The number of tracks was often proportional to the tape size that could be large up to 2" (50.4 mm) for 16 tracks.

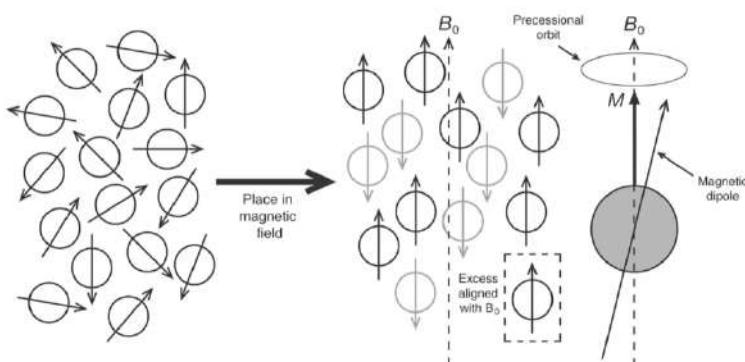
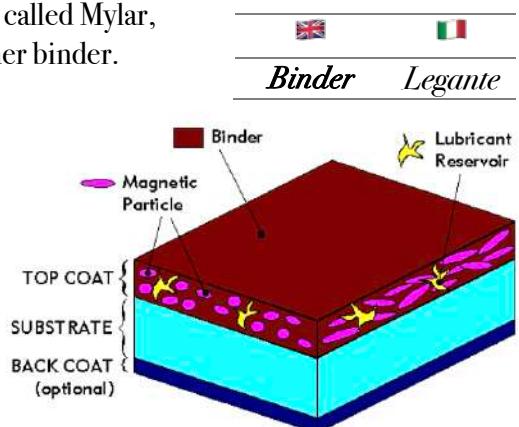
Once again, we have the problem of equalisation: different markets have different equalisation standards. The Europeans go with CCIR while the American/Japanese world uses the NAB. Furthermore, we can also use different types of coding: Dolby or DBX.

## How a tape works

Magnetic tapes are made of polyester film, in a synthetic material called Mylar, coated with a mix of magnetic pigment suspended within a polymer binder.

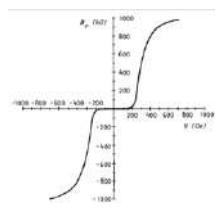
Everything happens in the topcoat; the substrate and the back coat are there just for mechanical reasons. The topcoat is constituted by a binder that embeds magnetic particles and lubricant reservoir that are needed to reduce friction and, consequently, variation in speed during the reproduction.

Let us start from the end: as tape post-processing phases we need to dry the tape, to smooth it (once again, to minimise the friction), to cut it and finally to pack it in a reel or inside a cassette. Let us see now how the magnetisation process works. We know that every magnetic particle can be easily



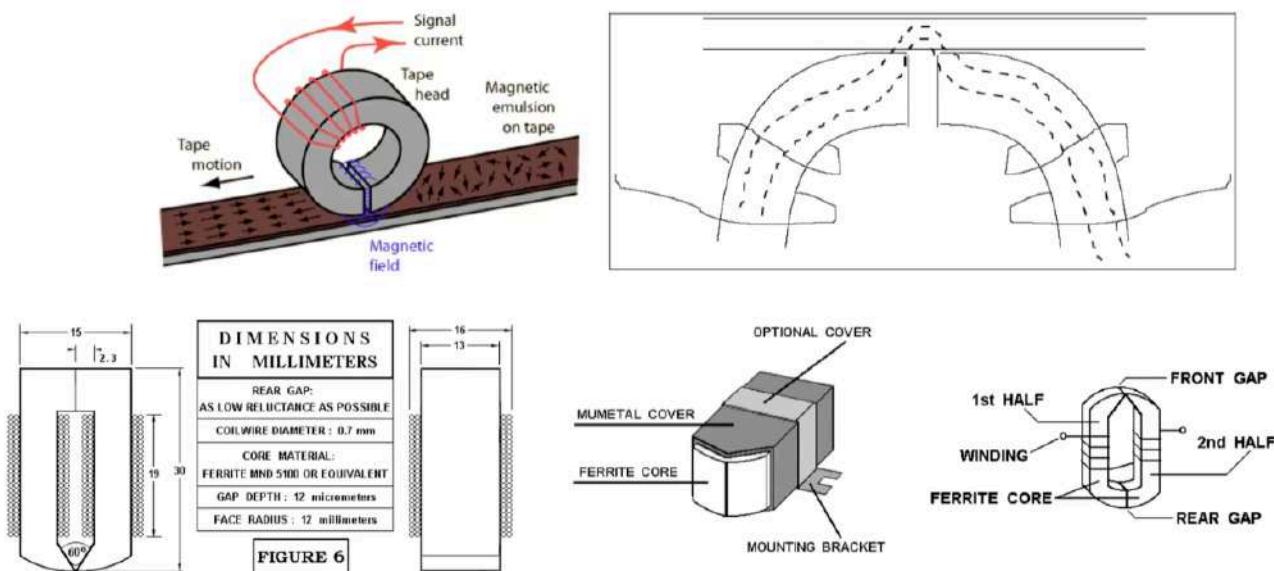
represented by a dipole. When these dipoles are oriented by a magnetic field, they acquire the orientation that the magnetic field itself has. This orientation is maintained, statistically, for a long time. Let us review why this happens. Exposing the magnetic material to an increasing magnetic field  $H$ , the residual magnetisation  $B_r$  varies according to a

particular curve called hysteresis curve. The maximum level of magnetisation depends on the magnetic characteristics of the adopted oxide, while the minimum level depends on the size of the oxide grains: the smaller the grains, the larger the minimal magnetisation. If the concentration of the polymer binder on the tape is low, due to deposition defects, we will have an amplitude modulation noise (drop out). Our aim is to reduce this noise.

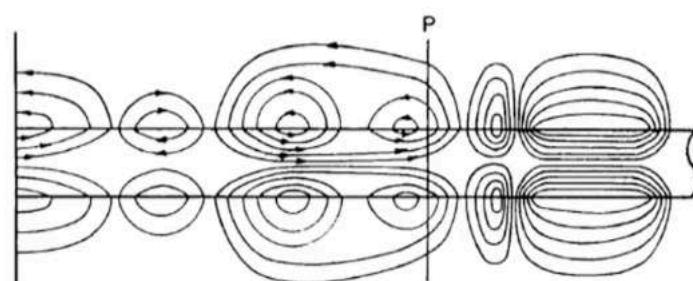
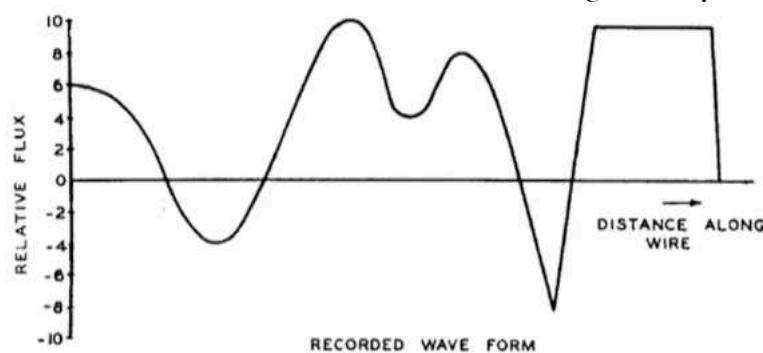


Before answering to this question, let us see how the magnetisation process is done.

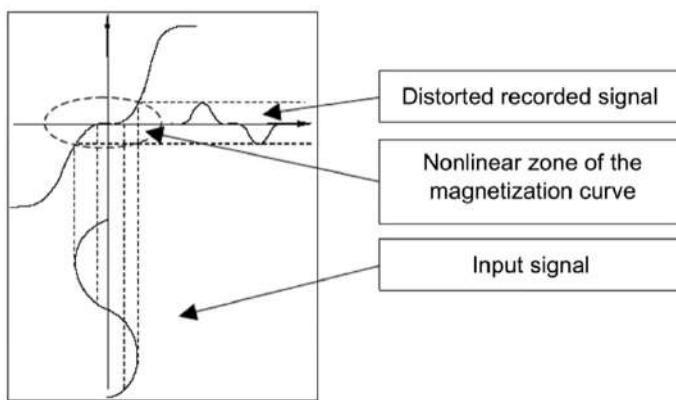
Everything is absolved by a magnetic circuit in which the signal current induces a magnetic field inside a ferromagnetic material. In correspondence of the contact with the tape, there is an air gap in the ferromagnetic ring. Since the magnetic field always follow the lowest impedance path, it will not flow through the air but through the tape: in this way, the dipoles will be magnetised according to the magnetic field direction.



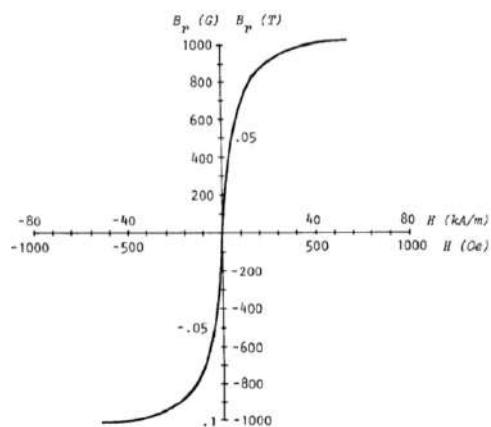
This is what we obtain, in terms of geometry of the magnetic field, after a magnetisation. As we can notice, when the flux is positive the lines rotate counterclockwise, when it is negative they rotate clockwise.



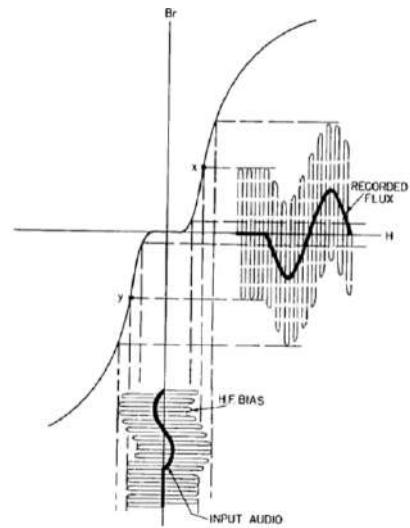
## Distortion



signal that goes from 50 to 400 kHz, depending on the oxide. This signal shifts the working point in the linear region of the characteristic.



Since we are interested in the crests and not in the zero crossing, we are done. All we need to do is to chase the crests of the signal that we retrieve by cutting the halfway below (using a diode) and apply a low pass filter. We implement a crest follower. At the end, what we obtain is a linear characteristic like the one reported on the left.



## Tape Speed

The simple rule that we follow to regulate the tape speed is  $F = \frac{v}{l}$  where  $v$  is the tape speed,  $L$  is the gap size (of the magnetic head) and  $F$  the maximum recordable frequency. A typical speed for professional tape recorders is  $19 \frac{\text{cm}}{\text{s}}$ . With a gap of  $9.5 \mu\text{m}$ , this enables a max bandwidth of 18 kHz. For compact cassettes of  $4.75 \frac{\text{cm}}{\text{s}}$  with a  $2 \mu\text{m}$  gap, if we want to guarantee a frequency response that does not vary more than  $\pm 2 \text{ dB}$  between 40 Hz and 15 kHz, we need to double the tape speed.

As for dynamic range:

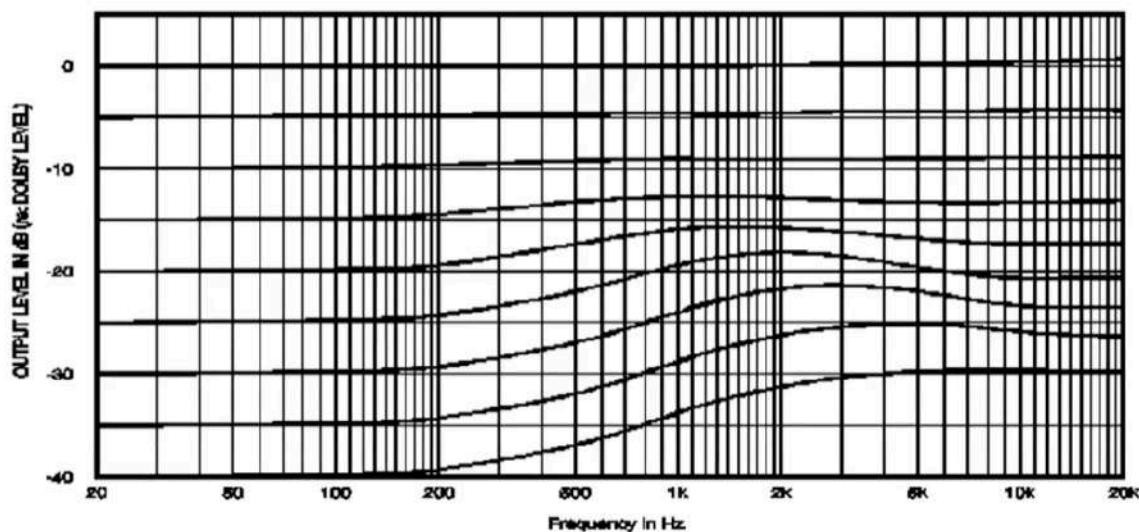
Type of tape	No. of tracks	Speed (cm/s)	Dynamic range (dB)
$1/4$	2	38	60
$1/4$	2	19	54
$1/4$	2	9.5	48
$1/4$	2	4.75	42
$1/8$ Fe per CC	4	4.75	40
$1/8$ Fe super CC	4	4.75	52
$1/8$ Cr per CC	4	4.75	54
$1/8$ Cr super CC	4	4.75	58

As we can notice the dynamic range is very little if we think about the fact that an orchestra of 75 elements has a dynamic range of 115 dB. Some solutions arise.

## Dolby

This technique was proposed by R. Dolby in 1983, it is used for reducing wideband noise (hiss). We underline that the hissing noise is generated by playing the tape: it is not in the recording as it is generated by the impulses that come from the magnetic elements in which the information is stored. You cannot remove it since are those impulses that generate the sound. The principle is similar to the magnification of high frequencies in RIAA equalization for LP records, except for the fact that the magnification is inversely proportional to the level of the recorded sound. For loud sounds (high SNR) the noise reduction is not necessary. In fact, performing it would lead to distortion problems and a loss of fidelity. We have to remind that Dolby alterates the signal, so it is better not to use it when it is not needed. There are two different type of Dolby

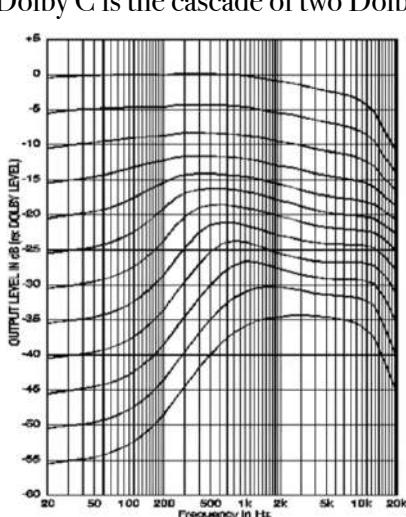
- Dolby B coding increases the high frequencies in the recording phase for a maximum of 10 dB for the weakest sounds (and reduces the same frequencies of the same amount in the playing phase).



We underline that this is a particular type of filter as its response depends on the strength of the signal: as we can notice, the lower the power of the signal, the highest the magnification on the highest frequency. Only the weakest sounds are high-pass filtered.

In this way, we emphasize the sound in which the hissing noise will appear and, when reproducing, we deemphasize those same frequencies back to their original values along with the noise itself.

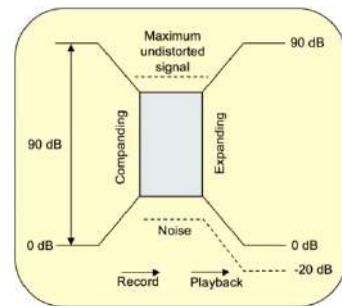
- Dolby C is the cascade of two Dolby B systems, and it enables a reduction of the white noise of about 20 dB. As we can notice, this time, high frequencies are also deemphasized. These curves came up after a lot of experimental attempts. By using the Wiener-Filtering technique, we can use theory to be more precise exploiting statistical information to shape the filters.



## DBX compressor/expander

Another technique that was used a lot, is the DBX. Its aim is to improve the dynamic range of the system. During the seventies, it was usual to see a box placed over the amplifier: it was the DBX! A vinyl record and a magnetic tape do not go beyond 50dB, regarding the dynamic range. In music, a climax from *pppp* to *ffff* covers more than 90 dB. We have to find a way to compress such a range into 60 dB.

This is done by means of a non-linear curve, but this comes at the cost of not properly reduce the hissing noise. The higher the expansion, the higher the noise.



## Impulse noise in tapes

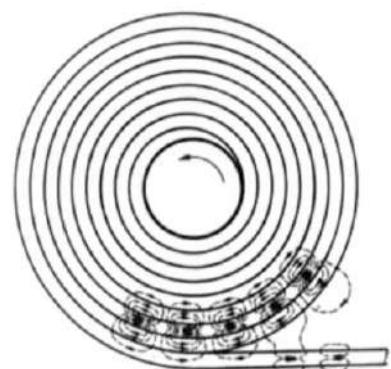
Let us see now which is the noise that we can encounter when we deal with tapes. One of the reasons could be due to local warping of thermal or mechanical origin (for examples someone that fixed the tape). It is quite different from scratches in records as it has no after-oscillations, a smaller amplitude, a critical phase of much longer duration. This affects the spectral content at high frequencies, which make the disturbance audible even when its magnitude is modest. Like in the case of scratches it is necessary to reconstruct the lost samples with an interpolation from surrounding areas where the signal was untouched. How can we remove it automatically? By exploiting once again statistical process.

Let us see which are the degradations that we can encounter in tapes:

- Degeneration of the binder
- Loss of lubricant
- Magnetic particles' instability
- Substrate warping
- Print-through

Let us see in which this last problem consists: the print-through, nomen omen, is an undesired magnetic transferal from one portion of the tape to the next wended layer in the same reel. It sounds like an “echo” of the previous recording portion or like a “precursor” of the next one, both at a very low level (50-60 dB lower). We list some of its characteristics:

- It increases with the temperature
- It increases with contact duration
- It is greater with thinner tapes
- It can be reduced by fast forwarding or rewinding the tape quite often
- It increases in the presence of magnetic fields | It is more evident at lower frequencies



## Phonographs recording - A little bit of history

Let us now switch to Phonographs.

In 1877: Thomas A. Edison invented the cylinder phonograph. It was able to record sounds by transferring pressure waves directly into a groove carved into an aluminium sheet that covers a rotating cylinder.

1887: Emil Berliner's flat record.

1925: Berliner's first non-acoustic recording. It was based on the use of the first microphone and of the “vacuum tube” for the conversion of sound into an electrical signal. The generated and amplified signal was, in fact, used to control the recording needle that carved the record's grooves. Here, we can hear the voices of some of the most famous people in history: <https://www.youtube.com/watch?v=etnXNSrSFEA>

1929: Arthur Keller introduces the multi- carrier modulation

1931: Sawyer introduces the crystal- based acoustic player

## **Phonographs recording – history of formats**

### **1877 – Phonograph Cylinder**

Thomas Edison's first demonstration of audio recording using a tin foil wrapped around a cardboard tube: <https://youtu.be/PRJpn02crbI>

### **Graphophone/Dictaphone Cylinders – 1887 (Volta Lab by Alexander Graham Bell in Washington, D.C.)**

They used lateral groove movement recording (like in LP records). In 1907, cylinders were renamed Dictaphone cylinders (which could be shaved and re-recorded over).

### **Ediphone – 1888**

Thomas Edison's Ediphone was in direct competition with the Graphophone. Edison originally used tin foil for his recordings, but he switched to wax in 1888. Unlike the lateral recordings (side-to-side groove carving) of the Graphophone, the Ediphone utilized a 'hill and dale' recording (vertical movement).

### **Brown Wax Cylinders – 1889**

Released in the late 1880's. They were the first mass market cylinder.

Most of the commercial recordings up the mid-1890's were recorded directly from a live performance. When purchasing a cylinder customers would therefore receive an original recording. As brown wax improved, many cylinders had the ability to be played up to 100 times before wearing out. This was a vast improvement over previous cylinder.

### **Pathé Cylinders – 1894**

Introduced by the French Pathé Freres Company to replace the current Brown Wax cylinders. Although they initially used brown wax, they moved to a black wax in 1903. The cylinders came in various diameter sizes (2 1/4", 3 1/2" and 5"). Recorded from master recordings. Sold until 1914.

### **Grand Graphophone Cylinders – 1898**

In 1898, Columbia introduced a new luxury format of cylinders with a diameter of 5". Although it did not provide a longer playing time, there was much more recording surface which allowed for improved sound. This format ultimately failed due to its higher cost.

### **Gold Moulded Records – 1902**

Similar to the French Pathé Cylinders in that they were made from a hard black wax. Edison was able to create a mould from a master cylinder that allowed him to produce several hundred cylinders from a single mould (the master cylinder was coated with a thin layer of gold). Discontinued in 1912.

### **Indestructible Record – 1907**

The Phonographic Co of Albany, NY starting started the production of the "Indestructible Record" in 1907. Made of celluloid, they were much more durable than other formats. Discontinued in 1922.

### **Amberol Records – 1908**

They were an improvement over Gold Moulded records as they had twice as many grooves, which allowed for longer playing times (4 minutes). The Amberol wax was harder than the Gold Moulded wax. They did have a tendency to crack or shatter rather easily and wore out rather quickly. In 1912 they were replaced with the Blue Amberol Records, which were made of blue celluloid around a plaster of pairs core and could be played hundreds of time with limited noise. Edison Records closed in 1929.

### **1894 – GRAMOPHONE RECORD – Analog, Mechanical/Horizontal stylus motion**

Lateral-cut disc records were developed by Emile Berliner in 1894, who named his system the "Gramophone". During the time of development, records varied in size and speeds. By 1925, the speed of the record was becoming standardized at 78 rpm. Various materials had been used to make the records, including rubber, but the standard ultimately became shellac. In the picture is an Edison Diamond Disc which was in production from 1912 to 1929. This record was unique in that it had a vertical stylus motion and played at 80 rpm.

#### ***1928 – VICTORY RECORDS – Analog, Mechanical***

In 1923, Woolworth commissioned two suppliers, Crystallate and Vocalion, to make records for its stores. As the offer grew in popularity it became very lucrative for the suppliers, but both feared that Woolworth might drop them and move to a single partner. In secret they hatched a plan to join forces and then pool resources to convert from a mechanical to an electronic recording process. On electrical recordings, a microphone was used to convert the sound into an electrical signal that was amplified and used to actuate the recording stylus. This eliminated the "horn sound" and produced a clearer and more full-bodied recording by greatly extending the useful range of audio frequencies and allowed previously unrecordable distant and feeble sounds to be captured. The discs would have a seven-inch (17.5cm) format with clearer sound and a longer playing time. Woolworth was happy to accept the new arrangement. 'The Victory' releases included the latest popular songs and high-fidelity instrumentals, as well as dances and a popular children's series. The sixpenny (2½p) discs were unbeatable value, matching the 'Broadcast Records' which sold in specialist stores for one shilling and threepence (6¼p), for both quality and variety. In 1931, they were ultimately replaced in Woolworth stores by the Eclipse 8-inch 78rpm record.

#### ***Late 1920's – ACETATE DISCS***

An acetate is a metal disk covered in a cellulose acetate material or black nitrocellulose lacquer and were used in studios for approval before pressing to vinyl. In addition to the usual central spindle hole, there is traditionally at least one drive hole in the label area, meant to be engaged by a special pin that prevents the disc from slipping on the turntable during the recording process on lathes that don't have a vacuum turntable. Originally acetates were used for home recording in the days before tape recorders. A machine actually cuts the grooves into the acetate like a lathe. Many acetates could be off-air radio recordings of rare or unavailable radio shows or sample/test pressings of alternate takes of original masters of the songs.



#### ***Early 1930's – ALUMINUM RECORDS / VOICE RECORDS***

Aluminium recordable records were roughly the size of a modern-day CD. There were recordings on both sides, but one side was non-recordable and came with a pre-recorded advertisement (normally for cigarettes). You would find these machines in department stores or other public places and allowed for approximately 1 minute of (usually rather poor quality) recording.



#### ***1940'S-1980'S – CARDBOARD RECORDS***

Phonograph records that were basically a plastic-coated card. The first examples appeared in the 1940s and were played at 78 rpm. The audio quality of the cardboard records was quite poor and were often only good for a few plays. They also had a tendency to easily warp. Used in advertising campaigns, music promotions, personal recordings, greeting cards and postcards, and pretty much anything else you could use recorded sound for.



This is a recordable Packard-Bell PhonOcord record. This was one of many manufacturers of paper records for consumer machines. They also made machines as attractions at amusements areas for the general public to use.

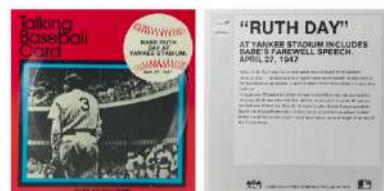
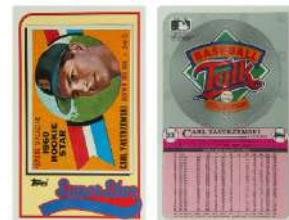
Below are cardboard records that were promoted on the back of cereal boxes. Originally manufactured on Wheaties boxes in the 1950's. You would simply cut them out along the dotted line and play them on your record player. Next to them on the right, a german carbord record embossed on a postcard.



Phonograph records that were basically a plastic-coated card. The first examples appeared in the 1940s and were played at 78 rpm. The audio quality of the cardboard records was quite poor and were often only good for a few plays. They also had a tendency to easily warp. Used in advertising campaigns, music promotions, personal recordings, greeting cards and postcards, and pretty much anything else you could use recorded sound for.

These are playable on the "Sports Talk Player", a unique audio player. There are interviews/highlights etc. The narrators were Don Drysdale, Joe Torre, and Mel Allen. This Carl Yastrzemski card is from a set of 164, which were originally issued with 4 cards per pack.

On the right, is a sports-related cardboard record known as a Talking Baseball Card. This one was released by CMC in 1989. It's a record of Babe Ruth's final speech from 1947 at Yankee Stadium.



## 1946 - RECORDING DISCS

Recording discs were released around 1946. Due to WW2 and their initial high cost, they did not really gain steam until after the war. At that time, home recorded discs were the best option for recording family and other events. Due to the eventual release of magnetic tape and the fact that acetate discs were inherently less durable not being physically editable, the records gave way to the tape. The example here is an unrecorded Silvertone Metal Core record.



## 1950'S - FLEXI DISC

The Flexi Disc was made from thin flexible vinyl. They were often seen in magazines or other printed publications where they could be mounted in the seam. They normally came in 4" or 8" sizes. This one came from the January 1979 edition of National Geographic.



## 1956 - HIGHWAY HI-FI - Analog, Mechanical

Developed to play in Automobiles. Chrysler introduced this into their automobiles in 1956. Manufactured by Columbia Special Products, it could hold approximately 45 minutes of music or one hour of speech per side. The 7" record spun at a slower rotation speed of 162/3 rpm. To avoid skipping, the stylus was pressed on the record with unusual pressure.



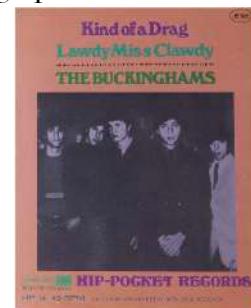
### **1961 – JUKEBOX EP (Little LP)**

These were small holed 7" records that played at 33⅓ rpm and were basically cut down versions of full albums. They even carried the same album artwork. It was thought that these would be a good alternative for jukebox records, but they ever gained much popularity.



### **1967 – HIP POCKET RECORD – Flexidisc**

Small 45rpm, 4-inch flexidisc records that were manufactured by Philco. Philco had also produced portable players for these discs, but they could be played on manual phonographs as well. The disks were sold in vending machines for 50 cents or counter displays at stores for 49 cents. They were small enough to be carried in a person's pocket or shipped in an envelope since they were flexible and not as fragile as a standard record. Many major record labels such as Columbia Records, RCA Records, Motown Records, and MCA Records chose not to participate. Furthermore, they were limited to 3-5 minutes of music, therefore longer songs (e.g. Beatles, Hey Jude) could not be played. Discontinued in 1969.



### **1978 – LASERDISC – Analog video, Digital or Analog audio**

Released in 1978. Home video format and first commercial optical disc storage medium. The standard home video LaserDisc was 12" (the size of a standard LP record) in diameter and made up of two single-sided aluminum discs layered in plastic. Although LaserDiscs capable of offering higher quality video and audio than VHS and Betamax videocassettes, the LaserDisc never managed to gain widespread use in North America, although were quite popular in Japan, and regions of Southeast Asia, such as Hong Kong, Singapore, and Malaysia. The technologies and concepts behind LaserDiscs were the foundation for later optical disc formats, including Compact Disc, DVD, and Blu-ray Discs. Many LaserDiscs contained music related material such as concerts, etc. Everything was recorder in digital modulation, so you had to read the zero crossing in order to reconstruct the signal.



### **1982 – COMPACT DISC (CD) – Digital, Optical**

The Compact disc (CD) is a digital optical disc data storage format that was introduced in 1982 by Philips. The Standard CDs have a diameter of 4.7 inches, is made of polycarbonate plastic and can hold up to about 80 minutes of uncompressed audio or about 700 MB of data. Although CD sales have continued to decline with the introduction of new formats, it remains today the primary source of recorded music.



### **2007 – VINYLDISC – Hybrid: Digital, Optical and Analog, Mechanical**

The VinylDisc (vinyl CD) is a hybrid CD and vinyl phonograph record. It was developed in Germany by Optical Media Production in 2007. The CD side is a standard CD while the vinyl side is a 33⅓ rpm record that can play up to 3.5 minutes of audio on a regular phonograph with use of a special adapter.

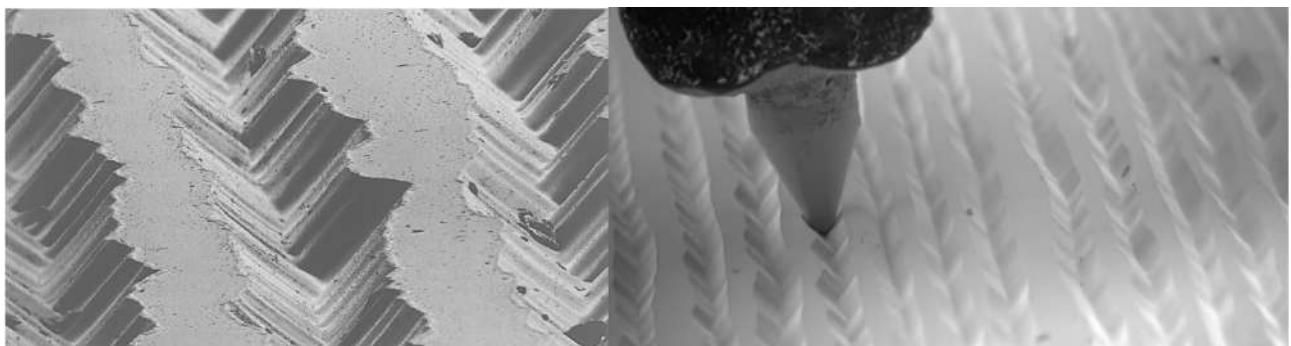
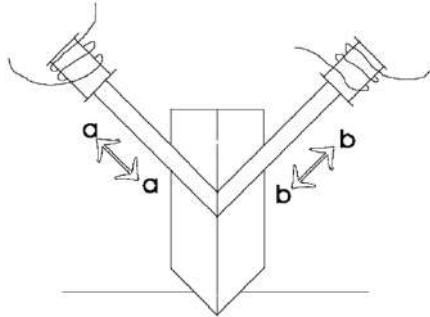


## Stereo recording - How it works on vinyl records

For backward compatibility, the two tracks are recorded on two skewed directions in a way that old mono record players read the sum of the two channels. If they used another type of configuration (e.i. vertical movement for left channel and horizontal movement for right channel) an old mono record player would have reproduced only the left channel neglecting the right one.

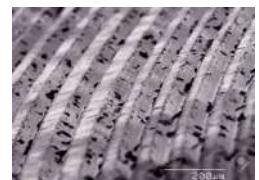
Since the two styluses are tilted, the signal has to be transformed in order to obtain the desired result according to the following rotation matrix.

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \cos(45^\circ) & \sin(45^\circ) \\ -\sin(45^\circ) & \cos(45^\circ) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$



## Degradation causes

Record's degradation is usually due to the granular nature of ebonite (old records) and vinyl (newer records), or to scratches. We have to remind that a vinyl is not a solid but a liquid, a very viscous liquid. Therefore, a grain of dust on its surface will sink in. The dust on the vinyl will become part of the noise when they are sunk as we can see in the picture. In general, granularity generates a background noise that is made of a high-density sequence of small and closely-spaced impulses ("crackling"). On the other hand, a scratch has a local impact and shows up as an impulse that has a steep transient followed by a mechanical oscillation (due to inertia, we have a little oscillation after an impulse). Therefore, during the initial sharp transient the signal is completely obliterated; during the following phase we can still adopt an additive signal+noise mode.



We dwell just a second on the difference between the crackling noise typical of vinyl and the hissing noise typical of tapes:

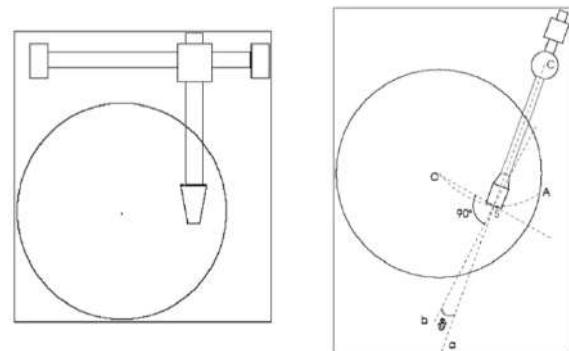
- Hissing noise: this type of noise is so dense, global degradation, as it comes from a series of impulses, that is perceived as a regular noise.
- Crackling noise: generated by "discontinuities", local degradation, is not dense enough to be perceived as a regular noise. We can perfectly distinguish each crackle in the sound.

This is why it is important to know which is the support of the original file while restoring. Hissing noise and crackling noise have to be dealt in two completely different ways.

## Rendering problems

There are also other types of issues that we have to take into account while with vinyl. The first one is the rumble noise generated by the vibration and the rotation of the turntable. It is useless to spend lot of money in a turntable that does not produce this kind of noise as it is impossible to remove them completely. It is better to spend money to create an environment adapt to music reproduction: a €50 000 system in the wrong room get the same result of a €200 system in the same place.

The second problem is the so-called tangential error. During the record phase, the needle always carves the grooves in a tangential fashion. On the contrary, during the playing phase the arm follows a circular trajectory. In order to avoid this problem, we should use tangential readers.

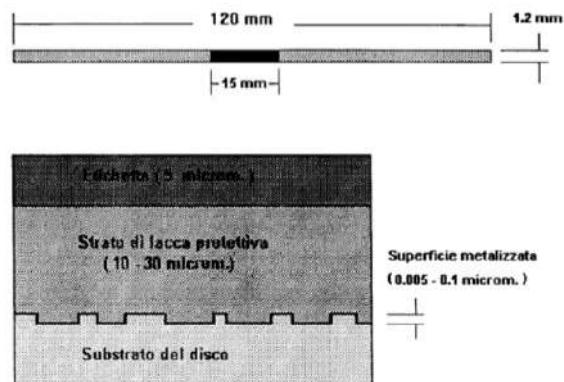
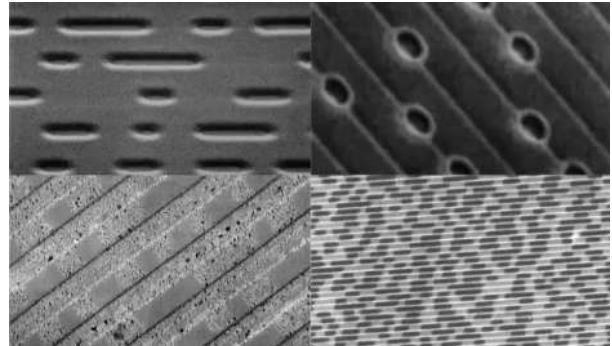


## Audio CD

The audio CD is the only device that has no physical contact with the machine that reads it. It was standardised in the mid-80s. Physically, it is an optical (typically read-only) storing media for digital audio signals. A sequence of "pits" are printed on the surface of a substrate of transparent plastic material (typically polycarbonate). This sequence carries the information. This surface is laminated with an extremely thin (50-100nm) layer of reflecting material (some metal based on aluminum, silver or gold). On top of this thin foil is a plastic protective layer (10 to 30µm thick).

Let us summarise its most important characteristics

- Pit width = 0.6µm
- Pit length depends on the binary sequence
- Distance btw two successive tracks = 1.6µm
- The information is hot-pressed on the disc
- The track spirals out from the centre of the disc toward the periphery
- The initial portion of the spiral contains the TOC (Table Of Contents), which is read and loaded by the player as soon as the disc is inserted
- The digital samples occupy the rest of the track as well as other information needed to localize the musical pieces and to correct the errors.



## Data reading

The reading process is a laser-based reading with no physical contact.

The laser beam goes through the plastic substrate and gets reflected by the metallic layer that covers the pit surface. The substrate surface where the pits are is physically separated from each one of the two external surfaces of the disc. Dust particles and scratches that are on the disc surface are not on the focal plane of the laser, therefore their impact is modest. Anything smaller than 0.5 µm is negligible and does not introduce any phase error in the reading of the audio CD.

Since we have to deal with rotational motion we perform a so-called constant-density reading: the revolution speed varies with the radial position of the reading head from an initial speed of 500 rpm to a final speed = 200 rpm.

An audio CD allows us also to perform error correction (missing or damaged pits due to writing errors or scratches) thanks to two techniques:

- EFM (Eight to Fourteen Modulation)
- RS-PC (Reed Solomon Product Code)

Such coding combination enables the correction of data loss corresponding to scratches up to 2.5mm wide (3500 bits). This means that a hole made with a little drill on the surface of the disc does not generate any perceivable problem.

There is only one thing that could affect the reproduction of a CD audio. This is the oily component that we have on our hand. That is why is important not to touch the CD surface directly.

## Digital Versatile Disc - DVD

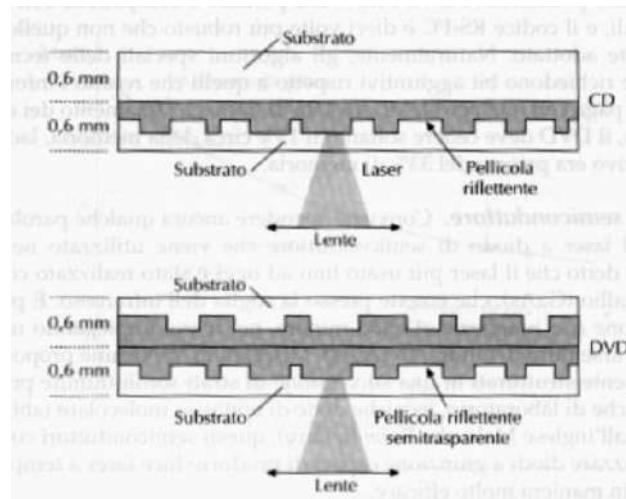
- It appears for the first time in 1995
- Smaller pits (higher density)
- Use of two overimposed layers for data storage
- Faster scanning
- More powerful error correction codes
- More effective laser focusing lens
- A greater data storage capacity: 8,5 GB vs. 0,68 GB of the traditional CD (12 times as much)

Let us see some differences between DVD and CD

Characteristics	CD	DVD
Diameter	120 mm	120mm
Thickness	1.2mm	1.2mm
Data layering	1 layer	1 or 2 layers
Length of a pit track	5Km	11Km
Separation between pit tracks	1.6µm	0.74µm
Track density	6250 ° tracks/cm	13500 tracks/cm
Minimum pit length	0.83µm	0.4µm
Wavelength of laser diode	0.78µm (near infrared)	0.635-0.65µm (red)
Numeric lens aperture	0.45	0.60
Pit scanning speed	1.25m/s	4.0m/s
Data storage capacity	0.68GB	4.7GB (1 layer) or 8GB (2 layers)
Data reading speed	153.6kB/s or 176.4kB/s	1108kB/s

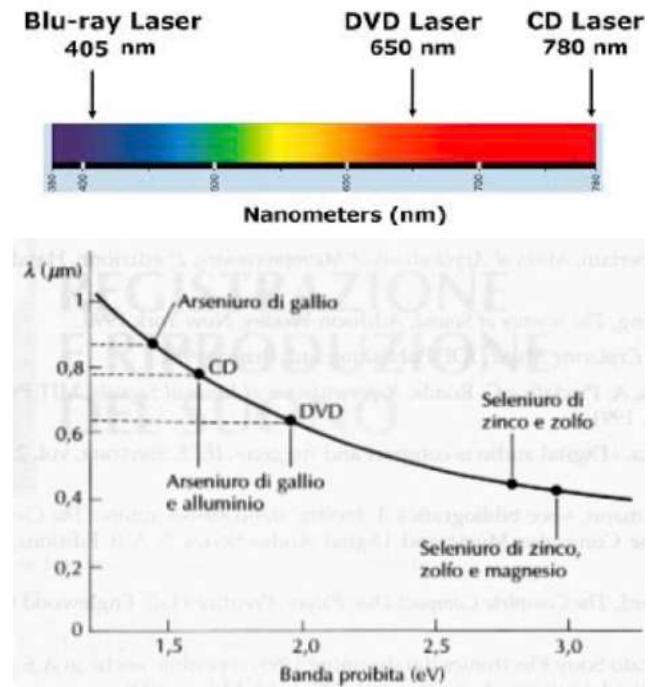
### Double layer

As we said, an advantage of the DVD is given by the double layer. The lower layer is semi-transparent and the lens can focus the laser beam either on the first or on the second layer.

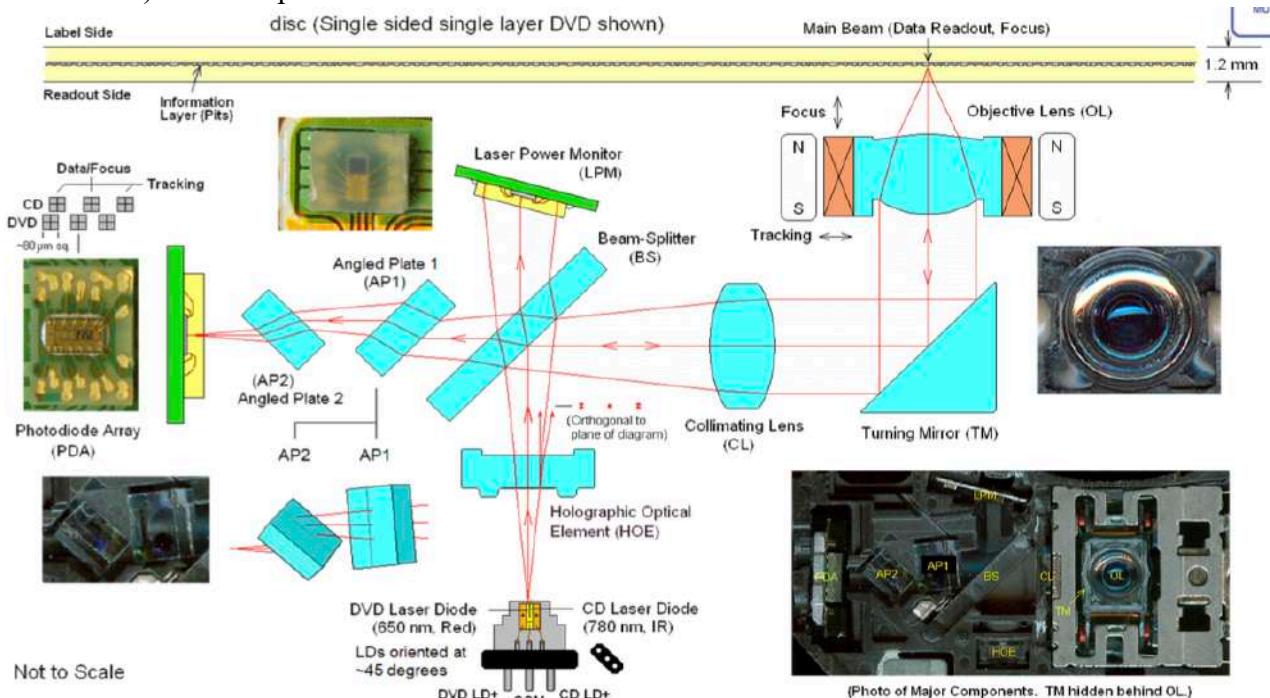


### Reading laser

In order to be able to read smaller pits, we need to keep diffraction at bay. This means picking a wavelength that is much smaller than the pits to read.



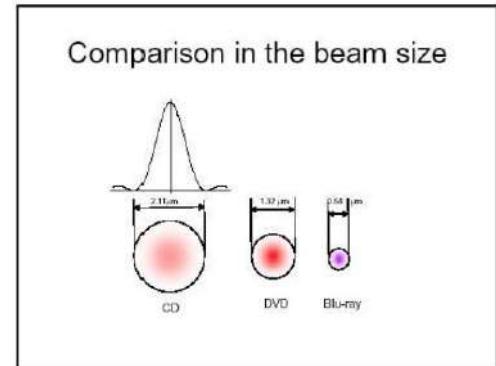
Before going on, let us take a closer look on the beam paths of an optical pickup (the device that read the disc) in a Philips/Lite-on DS-8A8SH CD/DVD RW Drive



## Blu-ray disc

Blu-ray discs were named after the laser used for data writing. The “e” is intentionally left out to trademark restrictions. The first blue laser was developed in 1996 by Shuji Nakamura (Nichia Corporation). In 2002, an alliance was formed (Blu-ray Disc Association), which includes Sony, Samsung, Sharp, HP and Philips. Let us see some of its characteristics:

- Single layer: 25 GB
- Dual layer: 50 GB
- Diameter: 120 mm
- Thickness: 1.2 mm
- Centre hole diameter: 15 mm
- Uses GaN laser of wavelength 400 nm
- The smaller laser, compared to the DVD and CD, keeps the process more efficient ( $\sim 5 \text{ mW}$ )



## Comparison of different disc characteristics

Type	BD	DVD	CD
Capacity (single layer)	25 GB	4.7 GB	700 MB
Capacity (double layer)	50 GB	8.5 GB	N/A
Wavelength of laser	405 nm	635/650 nm	780 nm
Numerical Aperture	0.85	0.6	0.45
Data transfer rate (1x)	36 Mbps	10.5 Mbps	1.17 Mbps
Thickness of the disc	1.2 mm	1.2 mm	1.2 mm
Recording method	Phase change	Phase change	Phase change
Data track	Groove recording	Groove (RW); Land (RAM)	Groove recording
Addressing method	Wobble	Wobble	Wobble

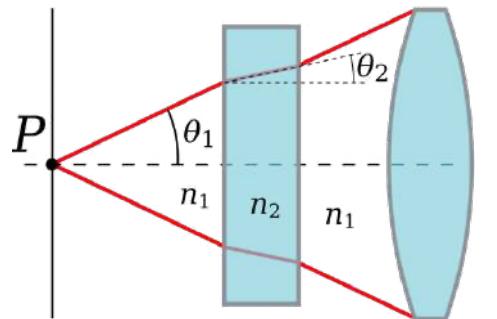
### Reminder: numerical aperture

Numerical Aperture measures the ability of a lens to gather and focus light. As the numerical aperture increases, the focusing power increases and the beam size decreases.

It is defined as

$$NA = n \sin(\theta)$$

Where  $n$  is the index of refraction of the medium.



### Reminder: phase change

The recording mechanism is based on the reversible phase transition between a crystalline and amorphous state of the recording material, induced by heating with a focussed laser beam. The reflectivity changes accordingly giving the bit representation.

### Reminder: groove/land and wobble addressing

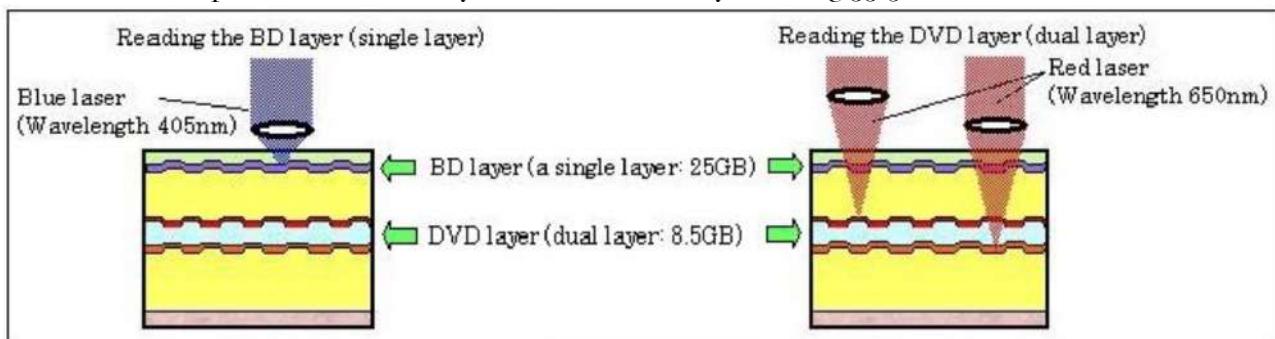
Optical discs, with the exception of DVD-RAM, have their data encoded on a single spiral, or a groove, which covers the surface of the disc. In the case of recordable media, this spiral contains a slight sinusoidal deviation from a perfect spiral. The period of this sine curve corresponds to the *wobble frequency*. The wobble frequency is commonly used as a synchronization source to achieve constant linear velocity while writing a disc, but has other uses as well depending on the type of disc. The frequencies quoted all assume that the disc is being written at the '1x' speed. The frequencies are appropriately higher for faster writing speeds.

## Recorder Characteristics

Over two hours of HDTV can be placed on a single layer BD, which correlates to over 13 hours for standard TV. The transfer rate is 36 megabits per second which means that, at ix speed, it takes approximately 1.5 hours to record an entire single layer BD.

## Compatibility

The issue of compatibility has introduced a competitor, the HD-DVD, that based its technology around being compatible with the DVD. Recently the BDA has developed recorders that are BD/DVD/CD compatible. JVC has advanced the BD by developing a BD/DVD combo disc that stores both DVD and BD data. It is composed of two DVD layers and a third BD layer storing 33.5 GB total.

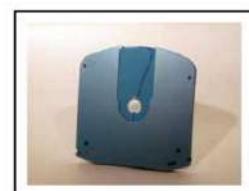


## Protection system

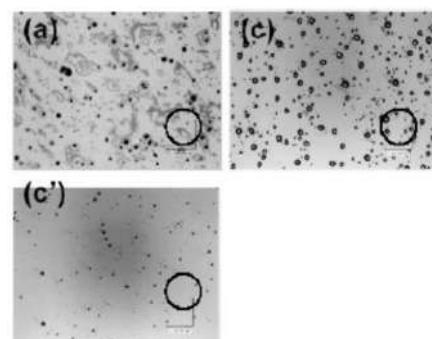
HDTV contains a copyright bit that is detected by the BD recorder. If the broadcast has no copyright bit, then the BD recorder is allowed to store the information. It uses the Data Encryption Standard (DES) that has a key length of 56 bits. A Key Block and Disc ID are written into the ROM area to prevent illegal copying.

## Enhancements

Error rates increased in reading and writing after the original BD suffered scratches and fingerprints. For protection, the prototype BD was enclosed in a case that increased the size of the disc drive.



To fix this problem, a hard coat was derived of an ultraviolet-curable resin that has a scratch resistance similar to the DVD. An artificial fingerprint reagent is placed on the disc surface to resist fingerprint oils



## Degradation in DVD and CD

With DVD or CD we have two forms of limiting factors:

1. The limited dynamic range to 16 bits even though we have to say that now there are some techniques that allows to overcome this limit.
2. Data Loss

## Digital Audio Restoration

We start our discussion highlighting the importance of sound restoration. We need to restore for mainly three reasons:

1. The expectations on sound quality grows as time goes by
2. Interest in historical material (old records, old tapes)
3. Noise reduction in contemporary digital recondensing

More generally, we restore to remove degradations. We can split the types of degradations in two different classes:

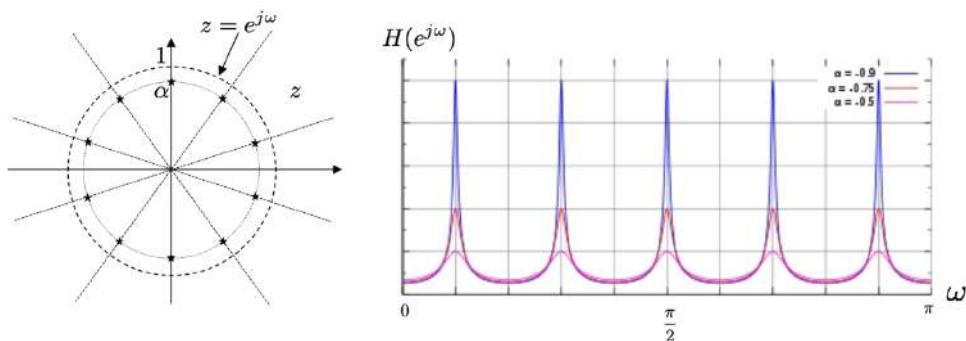
- Localised degradations: the one that can be spotted as discontinuities in the waveform which act only on certain samples (e.g. clicks, crackles, scratches, breakages, clipping, ...)
- Global degradations: deterioration spread on all samples of the waveform (e.g. background noise, wow and flutter, certain types of nonlinear distortion, ...)

Before starting, let us review a particular kind of signal called autoregressive signal (all-pole). Let us see the definition:

$$s(n) = \sum_{i=1}^P a_i s(n-1) + e(n)$$

Where  $P$  represents the order of the AR model and  $e(n)$  the innovation process. As we can notice, an autoregressive signal is a signal whose value can be represented by the previous samples of the signal at the time in which we are evaluating its value.

The AR model is a suitable representation for many stationary linear processes like noise-like signals (pole sclose to origin) and near-harmonic signals (poles close to the unit circle). A subgroup of the Autoregressive signal (AR) is composed by the so-called autoregressive moving average (ARMA) signals. This kind of signals are more flexible, but they need high model order ( $P > 30$  for simple signals) and to be more adaptive to cope with lack of stationarity. Below, the z-plane and the module of an autoregressive function is reported.



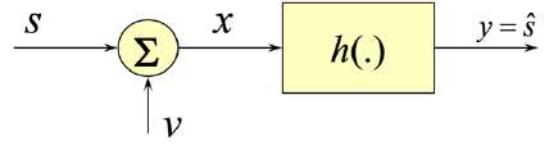
Another model that can be used to represent signals is the old-fashioned sinusoidal model. It is used for speech applications. This model, in a constrain form, called Short-Time Spectral Attenuation (STSA) is used for noise reduction

$$s(n) = \sum_{i=1}^{P_n} a_i(n) \cos\left(\int_0^{nT} \omega_i(t) dt + \phi_i\right)$$

It enables both frequency and amplitude modulation, as well as birth and death of individual components ( $P_n$  varies with time). Sadly, parameter estimation is difficult.

## Recap and in-depth analysis on Wiener Filter

Let us now recap what a Wiener Filter is in order to apply it to the problem we are facing. Let us suppose to have a signal  $s$  that is degraded, by adding a noisy signal  $v$  uncorrelated and stationary, in  $x$ . The Wiener Filter, given as input the degraded signal  $x$ , can estimate the original signal  $s$ .



Mathematically speaking, we want to find  $h(\cdot)$  |  $MSE = E\{(s - y)^2\}$  is minimised. We know that this happens when  $e = (s - y) \perp x$  according to the orthogonality principle. As we saw in the previous lessons this happens if and only if

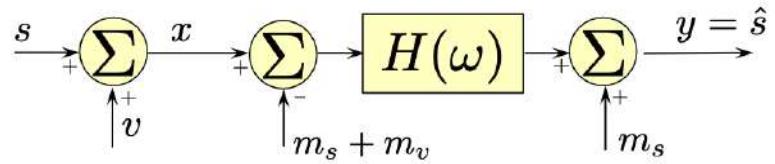
$$R_{sx}(n) = \sum_k h(k)R_x(n-k) \Leftrightarrow H(\omega) = \frac{P_{sx}(\omega)}{P_x(\omega)}$$

Since  $s$  and  $v$  are uncorrelated, we get  $P_{sx}(\omega) = P_s(\omega)$  and  $P_x(\omega) = P_s(\omega) + P_v(\omega)$ . The filter definition will become:

$$H(\omega) = \frac{P_s(\omega)}{P_s(\omega) + P_v(\omega)}$$

That guarantees the minimum mean square error between original and restored signals. The filter is a zero-phase low-pass filter in which  $P_s$  and  $P_v$  are assumed known or can be estimated.

But, what to do if the signal  $s$  is not zero mean? We simply make it zero mean by subtracting the mean value of the two signal and, after the filtering process, we add to the output just the mean of the signal  $s$ .



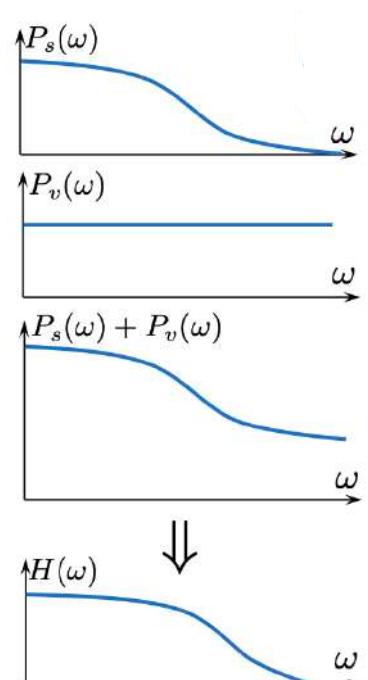
Let us see now how a Wiener filter can be represented on a Bode plot. In fact, we know that generally the SNR is lower for higher frequency and higher for lower frequency as the content of the signal is more relevant for lower frequencies and the noise spectrum is, generally, flat. It means that

$$\text{High SNR } (P_s \gg P_v) \Rightarrow H(\omega) = \frac{P_s(\omega)}{P_s(\omega) + P_v(\omega)} \rightarrow 1$$

$$\text{Low SNR } (P_s \ll P_v) \Rightarrow H(\omega) = \frac{P_s(\omega)}{P_s(\omega) + P_v(\omega)} \rightarrow 0$$

What we get, is the behaviour of a low pass filter. From these considerations we assume that:

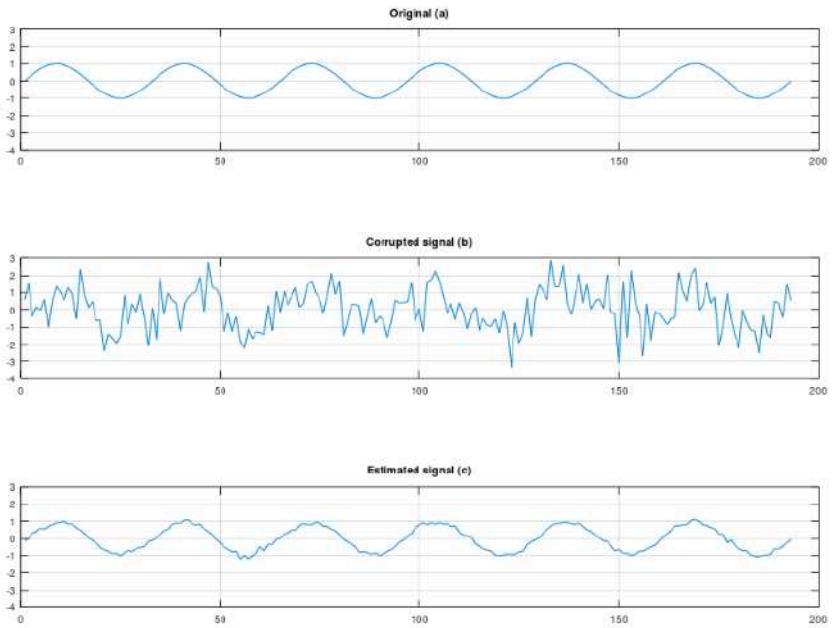
1. There is no point in filtering when the SNR is high
2. Better not do anything at all when the SNR is low



## Implementation and performance

For what it concerns the implementation, a Wiener filter is typically implemented in the frequency domain using DFT and IDFT. We remind that, as this implements a circular convolution, it corresponds to a linear convolution only if we first adequately extend zero-pad the sequences to account for wrap-around effects.

Regarding the performance we cannot not underline its strong effectiveness noise reduction. It is effective by definition as it is optimal in the MSE sense. But does this lead to a better perception in quality? Not at all! Given the fact that the system is inherently stationary, it has a heavy impact on sound quality (strong low-pass effect). The performance get better when it is used in adaptive processing.



## Characterisation

So, how can we characterise the effectiveness of a Wiener filter? During the theoretical discussion,  $P_s$  and  $P_v$  so far have been assumed as known but this is not so reasonable.  $P_v$  is relatively easy to estimate. We can hear the “sound of silence” in the recording, the portion of the track in which there is nothing but silence. There we can hear the background noise that can be assumed constant for the entire duration of the audio track. On the contrary,  $P_s$  is difficult to estimate. In some case it is possible to estimate  $P_s$  from clean samples that we assume having the same power spectral density. In other cases we can extract it from regions where noise is absent (if any).

Apart from particular cases, there are two main estimation approaches:

1. Periodogram

Given N frames  $s_i(n)$  of sound samples, we estimate the PSD as the average of the squared absolute values of the FFT of signal frames.

$$P_s(\omega) = \frac{1}{N} \sum_{i=1}^N |S_i(\omega)|^2$$

2. Model based estimation (Autoregressive models, MA, ARMA, ARMAX). They try to estimate the envelope of the signal.

### Reminder: Periodogram

The definition of Power Spectral Density (PSD) is usually given as the Fourier Transform of the autocorrelation of the signal (intended as a random process)

$$P_s(\omega) \triangleq \mathcal{F}\{R_s(n)\} \quad \text{where } R_s(n) = E\{s(k)s(k+n)\}$$

This, however, is not an operative definition. A more «practical» definition comes from Wiener-Хинчин theorem. Norbert Wiener proved this theorem for the deterministic case in 1930; Александр Хинчин formulated a similar result for stationary stochastic processes in 1934. Albert Einstein explained, without proofs, the idea in a brief two-page memo in 1914.

$$P_s(\omega) = E\{|S(\omega)|^2\} = E\{|S(\omega)|^2\}$$

The practical implications are in the approximations that can be introduced:

$$P_s(\omega) = E\{|S(\omega)|^2\} \approx \underbrace{E\{|S_w(\omega)|^2\}}_{\substack{\text{MS of the Fourier Transform} \\ \text{of the windowed signal}}} \approx \frac{1}{N} \sum_{i=1}^N |S_i(\omega)|^2$$

Therefore, the power spectrum of the signal can be approximated with a finite average of the square module of the Fourier Transform of all the frames. This is what we define periodogram: the average of the squared absolute values of the FFT of signal frames (frequency by frequency).

### Alternative formulation

Since we don't know the signal  $s(n)$ , but we do know  $x(n)$  and  $v(n)$ , the Wiener filter can be reformulated accordingly:

$$\hat{S}(\omega) = H(\omega)X(\omega) = \frac{P_s(\omega)}{P_s(\omega) + P_v(\omega)} X(\omega) \underset{P_s(\omega)=P_x(\omega)-P_v(\omega)}{=} \frac{P_x(\omega) - P_v(\omega)}{P_x(\omega)} X(\omega)$$

By retrieving  $H(\omega)$ , we get that

$$H(\omega) = \frac{P_x(\omega) - P_v(\omega)}{P_x(\omega)} = \frac{E\{|X(\omega)|^2\} - E\{|V(\omega)|^2\}}{E\{|X(\omega)|^2\}} \approx \frac{|X_w(\omega)|^2 - |V(\omega)|^2}{|X_w(\omega)|^2}$$

Notice that the numerator in the last expression runs the risk of becoming negative (random fluctuations), and this must be avoided (e.g. by clipping off negative values) because the numerator is expected to approximate a non-negative function (the PSD of  $s(n)$ ).

### Power spectral matching

Anyway, the Minimum Mean Square Error (MMSE) is not the only possible optimality criterion, nor is it necessarily the best (or the most relevant from the perceptual standpoint). An alternate optimality criterion comes from noticing that the Wiener filter does not lead to power spectral matching, as

$$P_y = P_x \frac{P_s^2}{(P_s + P_v)^2} = \frac{P_x}{P_s + P_v} \frac{P_s^2}{P_s + P_v} = \frac{P_s^2}{P_s + P_v} \neq P_s$$

Power spectral matching (PSM) can be achieved by defining

$$\hat{H} = \sqrt{\frac{P_s}{P_s + P_v}} \approx \sqrt{\frac{|X_w(\omega)|^2 - |V_w(\omega)|^2}{|X_w(\omega)|^2}}$$

as

$$P_y = P_x \hat{H}^2 = P_x \frac{P_s}{P_s + P_v} = P_s$$

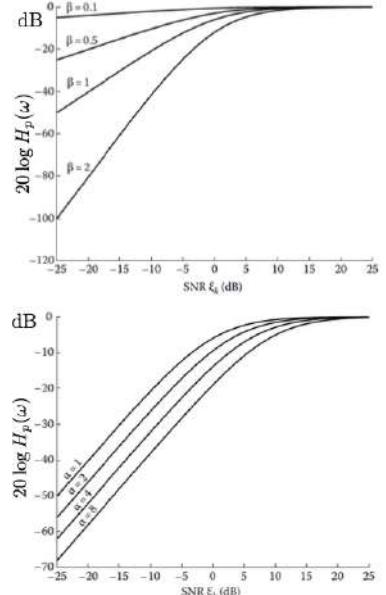
### Parametric Wiener Filtering

We can find a common ground between a classic Wiener filter, that minimises the MSE, and a Wiener filter that maximises the power spectral matching. This task is absolved by a filter called Parametric Wiener filter. A Wiener filter can also be written as

$$H(\omega) = \frac{P_x(\omega) - P_v(\omega)}{P_x(\omega)} = 1 - \frac{P_v(\omega)}{P_x(\omega)}$$

The Parametric Wiener Filter is a variation of the Wiener Filter that allows us to morph between Wiener filter and PSM exploiting the equation we have just written.

$$H_p(\omega) = \left( \frac{P_x(\omega) - \alpha P_v(\omega)}{P_x(\omega)} \right)^\beta = \left( 1 - \alpha \frac{P_v(\omega)}{P_x(\omega)} \right)^\beta$$



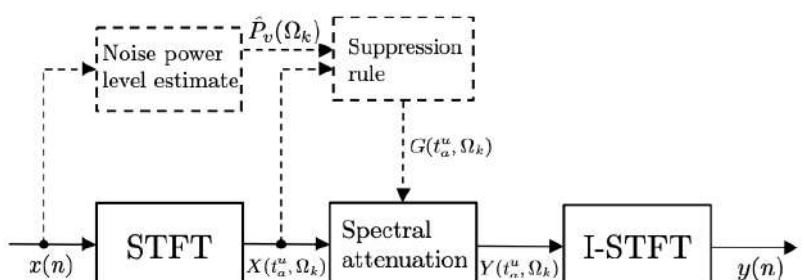
### Background noise reduction

Wiener filter can be used to reduce background noise. Noise model is assumed as additive (with signal and noise assumed as uncorrelated). Background noise (e.g. hissing) can be characterised as a single noise process that includes electrical circuit noise, irregularities of storage media, ambient noise from recording environment. It has components at all frequencies often is a white random process. But what happens if the signal is not stationary? Can we specialise the Wiener filter?

### Short-Time Spectral Attenuation

Simple stationary filtering (Wiener Filtering) is usually inadequate; therefore, we usually resort to its adaptive counterpart, known as Short-Time Spectral Attenuation. Let us see how the process works:

1. The analysis is based on STFT (or filterbanks)
2. The noise attenuation is performed on individual bands
3. At the end, the weighted bands are recombined.



Where the spectral attenuation function  $G(t_a^u, \Omega_k)$  depends only on the power level of the noisy signal measured at the same bin  $|X(t_a^u, \Omega_k)|^2$  and the power of the noise at the frequency  $k$  (assumed stationary)  $\hat{P}_v(\Omega_k) = E\{|V(t_a^u, \Omega_k)|^2\}$ .

Let us now dwell on the block called suppression rule. As we saw we can use three different suppression rules:

- Wiener Suppression Rule: 
$$G(t_a^u, \Omega_k) = 1 - \frac{\hat{P}_v(\Omega_k)}{|X(t_a^u, \Omega_k)|^2}$$
- Power Spectral Matching Suppression Rule: 
$$G(t_a^u, \Omega_k) = \sqrt{1 - \frac{\hat{P}_v(\Omega_k)}{|X(t_a^u, \Omega_k)|^2}}$$
- Parametric Wiener Suppression Rule: 
$$G(t_a^u, \Omega_k) = \left(1 - \alpha \frac{P_v(\omega)}{P_x(\omega)}\right)^\beta$$

What we have to notice is that in all of the above cases:

$$\lim_{SNR \rightarrow \infty} G(t_a^u, \Omega_k) = 1$$

$$\lim_{SNR \rightarrow 0} G(t_a^u, \Omega_k) = 0$$

## Wiener Filter and PSM filter implemented on MATLAB for noise reduction

Let us see an Example on MATLAB

```

1 % Sound Analysis, Synthesis and Processing
2 % Examples of the Wiener filtering. Time invariant approach
3 % Fabio Antonacci, February 2024
4
5 clear;
6 close all;
7
8 % Load the clean signal
9 [s, fs] = audioread('116-288045-0000.m4a');
10 time = (0:1/fs:(length(s)-1)/fs);
11 % Add the noise
12 sigma = 0.1;
13 v = sigma*randn(length(s),1);
14 x = s + v;
15
16 % Plot the noisy signal
17 figure(1)
18 subplot(3,1,1);
19 plot(time,x);
20 title('Noisy signal');
21 xlabel('Time [s]');
22 ylabel('Amplitude');
23
24 % Compute the power spectra
25 P_s = abs(fft(s)).^2;
26 P_v = abs(fft(v)).^2;
27 H_w = P_s./(P_s+P_v);
28
29 % Compute the fft of x
30 X = fft(x);
31
32 % Apply the Wiener filter
33 S_hat_w = X.*H_w;

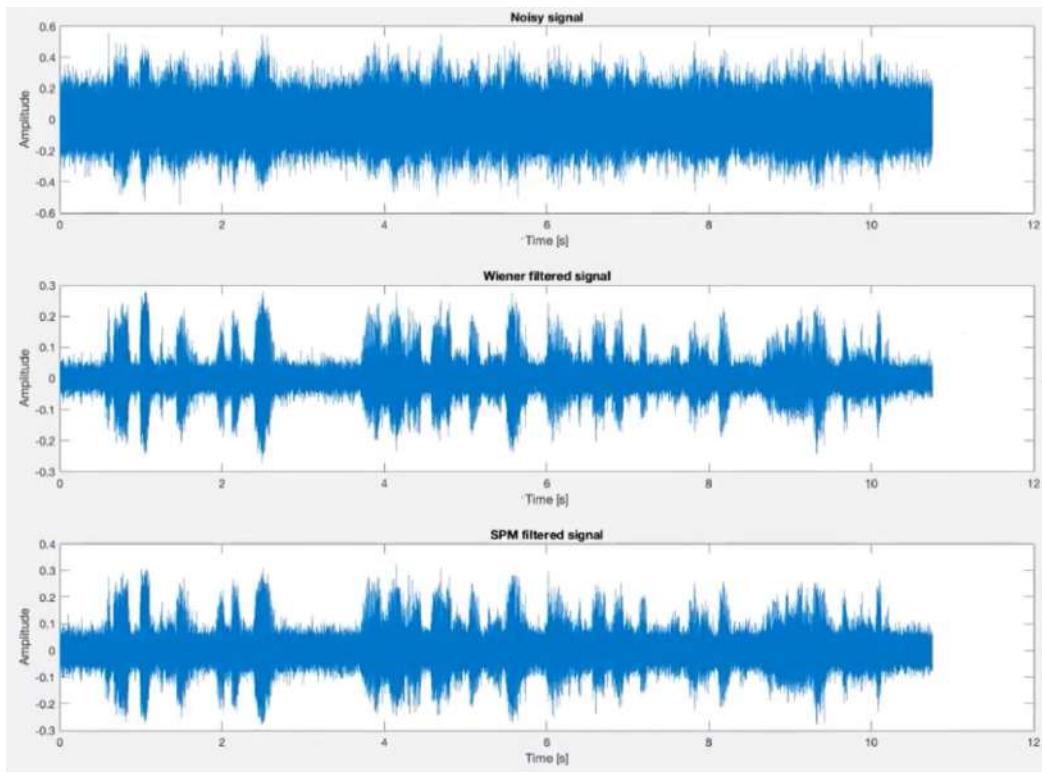
```

```

34 s_w = real(ifft(S_hat_w));
35
36 % Plot the Wiener filtered signal
37 subplot(3,1,2);
38 plot(time,s_w);
39 title('Wiener filtered signal');
40 xlabel('Time [s]');
41 ylabel('Amplitude');
42
43 % Apply the PSM filter
44 H_p = sqrt(P_s./(P_s+P_v));
45 S_hat_p = X .* H_p;
46 s_p = real(ifft(S_hat_p));
47
48 % Plot the PSM filtered signal
49 subplot(3,1,3);
50 plot(time,s_p);
51 title('PSM filtered signal');
52 xlabel('Time [s]');
53 ylabel('Amplitude');
54
55 figure(2);
56 subplot(4,1,1);
57 freq_axis = 0:fs/length(x):fs-fs/length(x);
58 plot(freq_axis,db(abs(P_s)));
59 title('Clean signal power spectrum');
60 xlabel('Frequency [Hz]');
61 ylabel('Amplitude[dB]');
62
63
64 subplot(4,1,2);
65 plot(freq_axis,db(abs(fft(x)).^2));
66 title('Noisy signal power spectrum');
67 xlabel('Frequency [Hz]');
68 ylabel('Amplitude[dB]');
69
70 subplot(4,1,3);
71 plot(freq_axis,db(abs(S_hat_w).^2));
72 title('Wiener filtered power spectrum');
73 xlabel('Frequency [Hz]');
74 ylabel('Amplitude[dB]');
75
76 subplot(4,1,4);
77 plot(freq_axis,db(abs(S_hat_p).^2));
78 title('Power Spectral Matching filtered power spectrum');
79 xlabel('Frequency [Hz]');
80 ylabel('Amplitude[dB]');

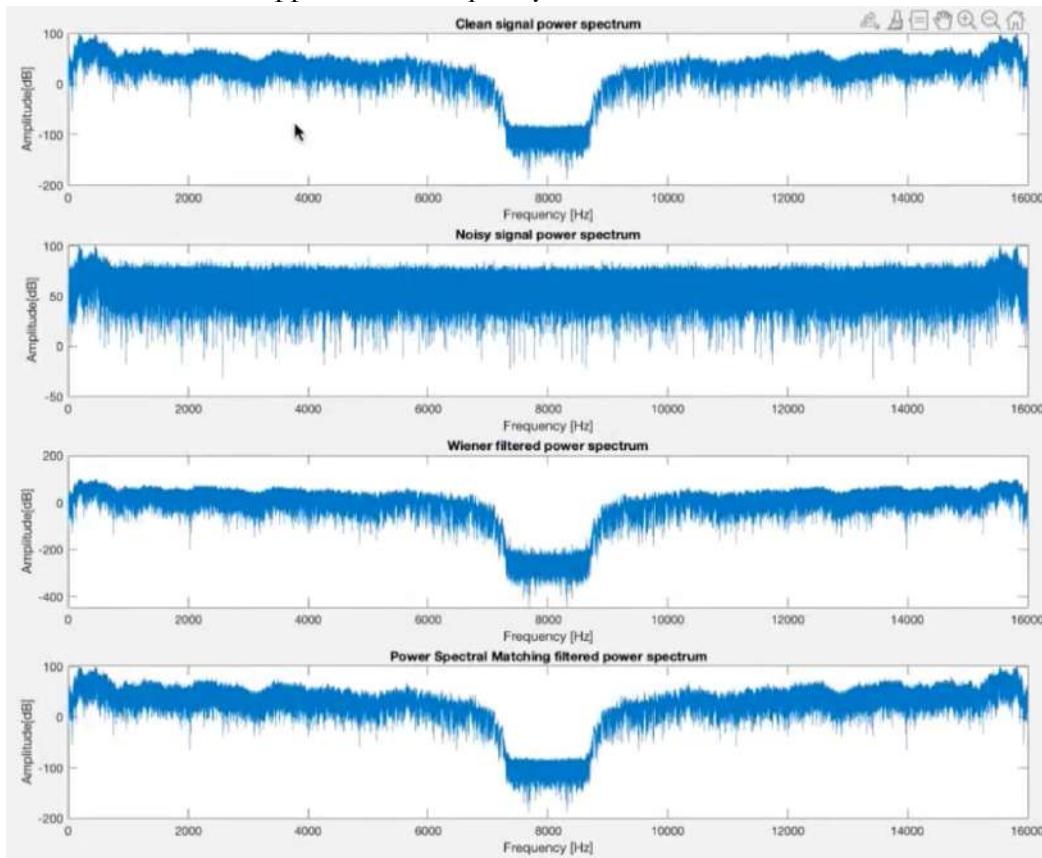
```

Let us see the results



On the top we have the nosy signal. We can notice that the Wiener filtering is more effective in removing the noise component with respect to the PSM filter. This is what we expected as the Wiener filter minimise the MSE. We highlight that this not necessary leads to a higher quality sound.

Let us see now what happens in the frequency domain.



We can notice how in the noisy signal power spectrum the target signal is completely covered by the noisy components. The original spectrum is retrieved by using a Wiener filter and a Power Spectral filter and, in both cases, the result is acceptable. By hearing to the signals, we notice that the noise reduction is more effective with

traditional Wiener filtering, while the preservation of the timber (of the spectral characteristics) is better preserved with the PSM filter. That is why we often use the parametric filter.

## STSA on MATLAB for noise reduction

Let us see now how things change by using a short time spectral attenuation filter.

```

1  %% Sound Analysis, Synthesis and Processing
2  % Examples of the Wiener filtering. Short time spectral attenuation
3  % Fabio Antonacci, February 2024
4
5  clear;
6  close all;
7
8  %% Load the clean signal
9  [s, fs] = audioread('116-288045-0000.m4a');
10 time = (0:1/fs:(length(s)-1)/fs);
11
12 %% Add the noise
13 sigma = 0.01;
14 v = sigma*randn(length(s),1);
15 x = s + v;
16
17 %% Plot the noisy signal
18 figure(1)
19 subplot(3,1,1);
20 plot(time,x);
21 title('Noisy signal');
22 xlabel('Time [s]');
23 ylabel('Amplitude');
24
25 %% Estimate the noise power spectrum through a silent portion of the signal
26 % (no speech)
27 [X,freq_axis,time] = stft(x);
28 idx = find(time/fs < 0.4);
29 P_v = sum(abs(X(:,idx)).^2,2)/length(idx);
30 P_v = smoothdata(P_v,1,'movmean',8);
31
32 %% Perform the filtering, frame by frame
33 % Wiener filtering
34 S_w = zeros(size(X));
35 for i = 1:size(X,2)
36     tmp = abs(X(:,i)).^2;
37     tmp = smoothdata(tmp,1,'movmean',8);
38     G_w = max(1-P_v./(tmp),0);
39     S_w(:,i) = G_w .* X(:,i);
40 end
41 s_w = istft(S_w,fs);
42 % Power Spectral Matching
43 S_p = zeros(size(X));
44 for i = 1:size(X,2)
45     tmp = abs(X(:,i)).^2;
46     tmp = smoothdata(tmp,1,'movmean',8);
47     G_p = sqrt(max(1-P_v./(tmp),0));
48     S_p(:,i) = G_p .* X(:,i);
49 end
50 s_p = istft(S_p,fs);

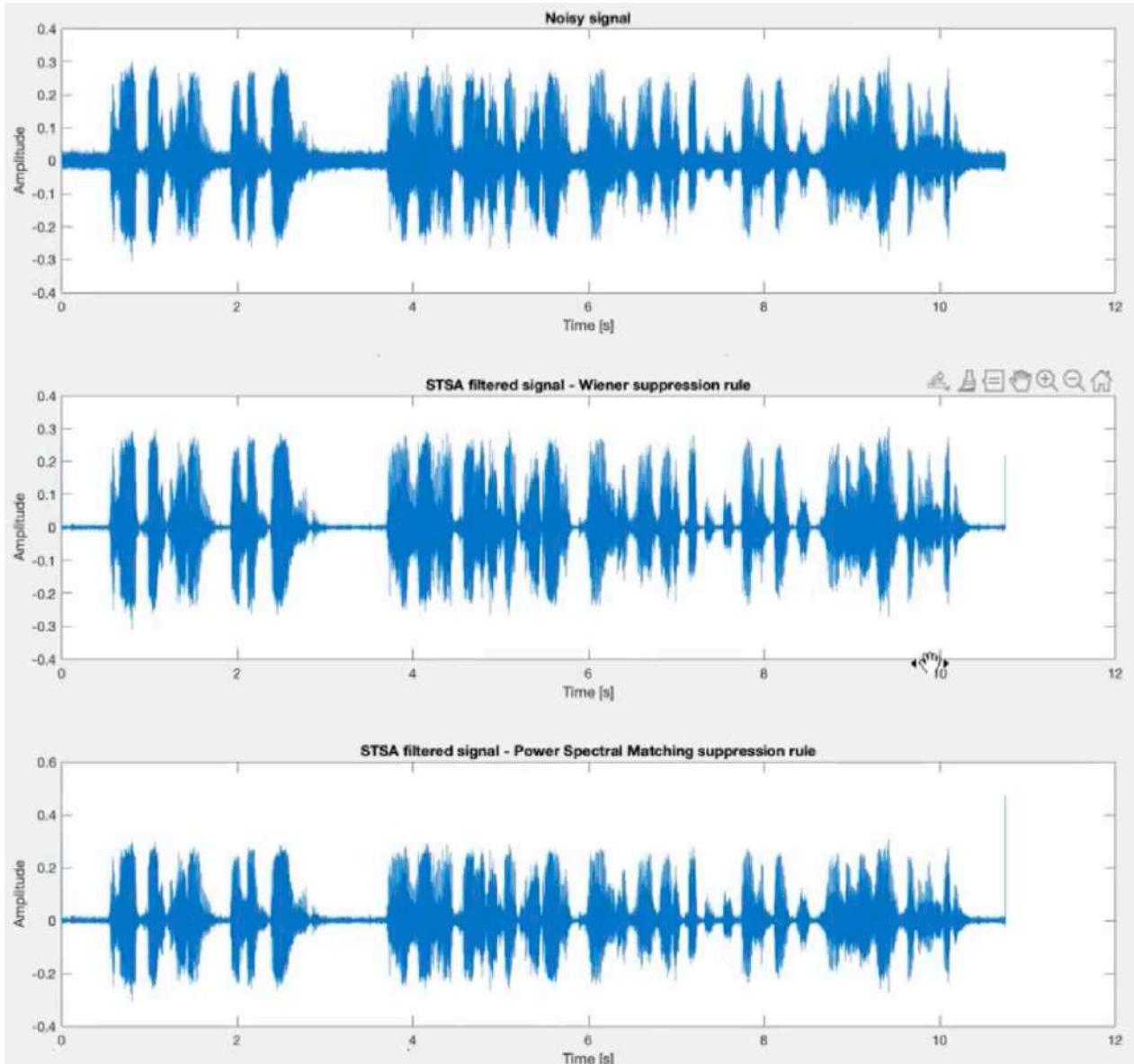
```

```

51 % Plot the STSA filtered signal
52
53 figure(1)
54 % Wiener suppression rule
55 subplot(3,1,2);
56 time_filt = 0:1/fs:(length(s_w)-1)/fs;
57 plot(time_filt,s_w);
58 title('STSA filtered signal - Wiener suppression rule');
59 xlabel('Time [s]');
60 ylabel('Amplitude');
61 % Power Spectral Matching suppression rule
62 subplot(3,1,3);
63 time_filt = 0:1/fs:(length(s_p)-1)/fs;
64 plot(time_filt,s_p);
65 title('STSA filtered signal - Power Spectral Matching suppression rule');
66 xlabel('Time [s]');
67 ylabel('Amplitude');

```

By looking at the result, it is noticeable how the noise reduction has been much more effective than before. The fact that the algorithm is specialised for non-stationary signals, returns a result that is more accurate.



## Dereverberation

In some applications (e.g. speech analysis) reverberations are considered a problem. This is a case of convolutive degradation source:

$$x(n) = h(n) * s(n)$$

The resulting signal  $x(n)$  is in fact given by the response of the environment (direct path + echoes) and the dry signal  $s(n)$ . How can we minimise its impact?

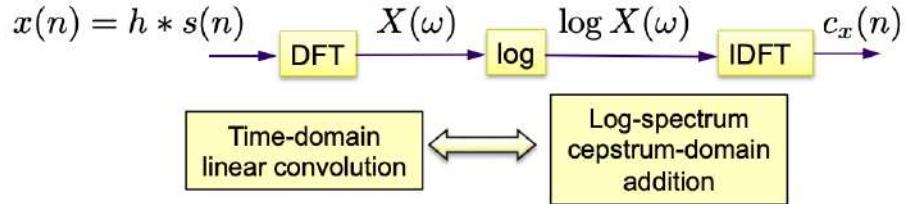
A general approach is to filter  $x(n)$  with inverse of  $h(n)$  following these steps:

1. Determine  $h(n)$ . This can be done by measuring  $h(n)$  or by estimating it from the speech  $x(n)$  performing what is called cepstrum based dereverberation.
2. Determine the “inverse” of  $h(n)$
3. Filter  $x(n)$  with the inverse of  $h(n)$

As we said, we have to use the cepstrum. But what is a cepstrum? When we analyse the spectrum of a signal, we notice that there are some parts that are rapidly varying with respect to the frequency and others that are slowly varying over frequency. The idea behind the cepstrum is to separate these two components.

If we were in the time domain, to separate a slowly varying component and a rapidly varying component we have nothing to do but to apply a filter. We try to do something similar in the frequency component. The problem is that in the frequency component we do not have a simple sum between a slowly varying component and a rapidly varying component but a multiplication. Things become more complex. We have to find a way to transform the multiplication into a sum. This can be done by applying a logarithm! The logarithm of a multiplication

is equal to the sum of the logarithms. Once we have done this operation, we can go back to the time domain and divide the two components. What we get is the complex cepstrum  $c_x$



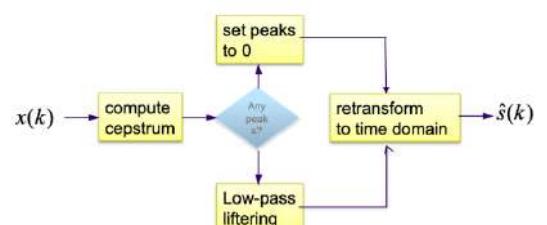
$$\begin{aligned} c_x(n) &= \mathcal{F}^{-1}\{\log(X(\omega))\} = \mathcal{F}^{-1}\{\log(H(\omega)S(\omega))\} = \mathcal{F}^{-1}\{\log(H(\omega)) + \log(S(\omega))\} = \\ &= \mathcal{F}^{-1}\{\log(H(\omega))\} + \mathcal{F}^{-1}\{\log(S(\omega))\} = c_h(n) + c_s(n) \end{aligned}$$

Once we have the spectrum, we can exploit some properties of speech and reverb signals:

- Speech is characterised by low-quefrency components in log-spectrum. So  $c_s$ , the signal will be concentrated around cepstral origin.
- The reverb channel is characterised by high-quefrency components in log-spectrum that are going to show a  $c_h$  made of pulses that are far away from the origin.

At the end, everything boils down to the following algorithm.

We compute the cepstrum of a signal given as input. We set to 0 all the peaks that are present in the cepstrum in order to avoid detrimental when going back in time domain. We apply to the cepstrum a low pass filtering (extracting the slowly varying coefficients, related to the speech) and, at the end, we retransform the signal in time domain.



When the room impulse response is known, we work differently. As we said, we simply have to “invert the convolution” that has the signal generated. The room impulse response (assumed as FIR) is contained in a vector called  $h[k]$ . But the inverse of a FIR is an IIR, and it is not guaranteed to be stable. To solve this problem, we compute an approximate FIR inverse as least squares solution of an overdetermined set of linear equations. Simply speaking, finding the solution that is closer to the ideal result in which we have an impulse just on the first sample of the vector (as it is reported on  $\underline{d}$ ).

$$\underbrace{\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ \underline{d} \end{pmatrix}}_{\underline{\underline{H}}} = \underbrace{\begin{pmatrix} h(0) & 0 & \cdots & 0 \\ h(1) & h(0) & \ddots & \vdots \\ \vdots & h(1) & \ddots & 0 \\ h(N_h) & \vdots & \ddots & h(0) \\ 0 & h(N_h) & \vdots & h(1) \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h(N_h) \end{pmatrix}}_{\underline{\underline{H}}} \underbrace{\begin{pmatrix} g(0) \\ g(1) \\ \vdots \\ g(N_g) \\ \underline{g} \end{pmatrix}}_{\underline{\underline{g}}} \Rightarrow \underline{g}_{LS} = (\underline{\underline{H}}^T \underline{\underline{H}}^{-1}) \underline{\underline{H}}^T \underline{\underline{d}}$$

As we can notice, as it is a Least Squared problem is solved by means of the pseudoinverse.

It can happen that if  $h$  is affected by some noise, the same noise is reported in the output. How can we solve this problem? By multiplying it for a coefficient  $\lambda$  small enough to seems zero (it affects the solution, but it solves the problem of conditioning).

He prepared a MATLAB example

```

1 % Sound Analysis, Synthesis and Processing
2 % Examples of dereverberation in the case of known RIR
3 % Fabio Antonacci, February 2024
4
5 clear;
6 close all;
7 % Create the signals
8 % Load the clean signal and the RIR
9 [s, fs_x] = audioread('116-288045-0000.m4a');
10 [h, fs_h] = audioread('BIG HALL E001 M2S.wav');
11 h = h(:,1);
12 h = h(1:10000);
13 figure(1);
14 subplot(311);
15 time_h = (0:1/fs_h:(length(h)-1)/fs_h);
16 plot(time_h,h);
17 xlabel('Time[s]');
18 % Resample the signal at the sampling rate of the RIR
19 h = resample(h,fs_x,fs_h);
20 title('Room Impulse Response')
21 % Convolve the dry signal with the RIR
22 y = conv(h(:,1),s);

23 %% Estimate the inverse channel
24 % Channel shortening of the RIR
25 time_cut = 0.2;
26 sample_cut = round(time_cut * fs_x);
27 h_c = h(1:sample_cut);
28 % Create the circulant matrix from the shortened RIR
29 H = convmtx(h_c,400);
30 % Impose the desired inverse filter
31 d = zeros(size(H,1),1);
32 d(1,1) = 1;

```

He reads an audio file containing the speech signal and he convolves it with the response of the big hall. He modifies the response to mono from stereo.

```

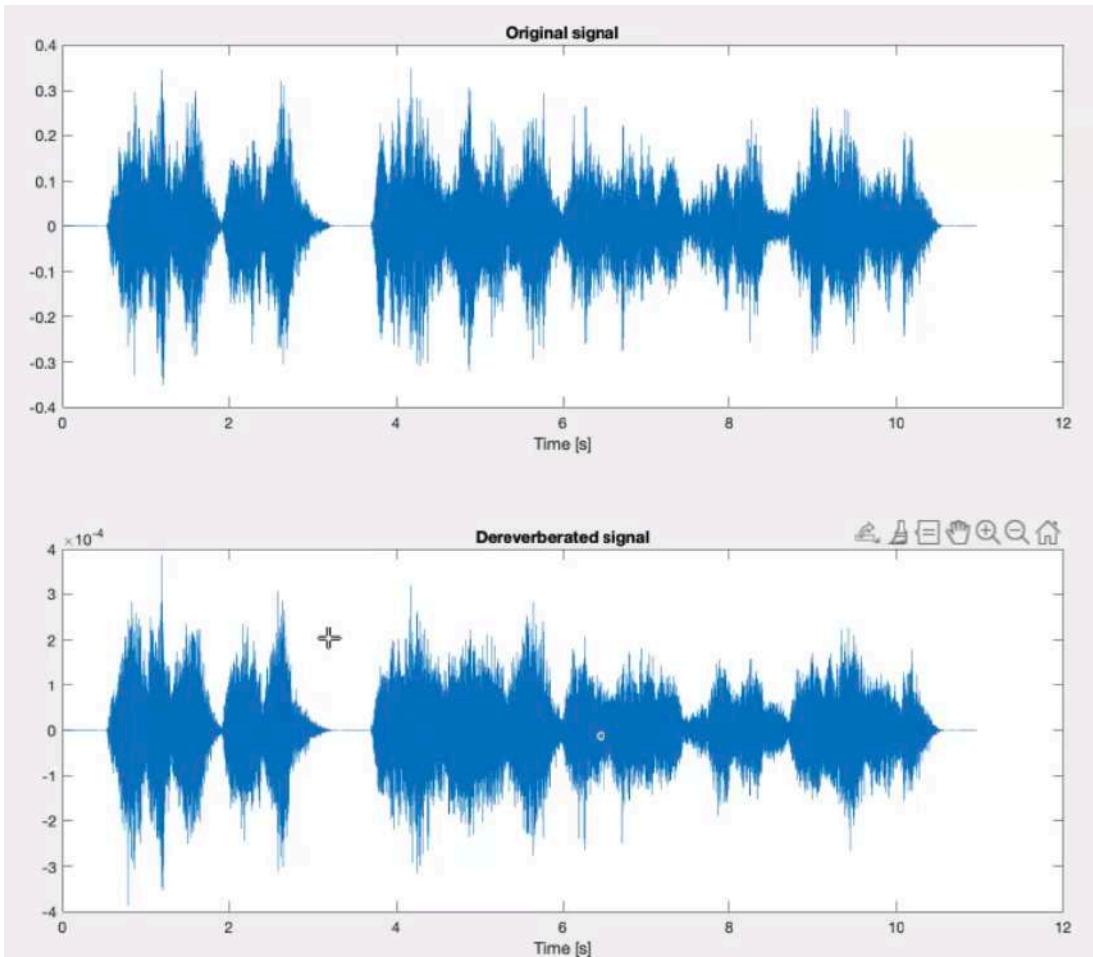
33 %% Perform channel inversion
34 g = pinv(H)*d;
35 subplot(312);
36 time_axis = 0:1/fs_x:(length(g)-1)/fs_x;
37 plot(time_axis,g);
38 xlabel('Time[s]')
39 title('Inverse channel');
40 subplot(313);
41 response = conv(h,g);
42 time_axis = 0:1/fs_x:(length(response)-1)/fs_x;
43 xlabel('Time[s]')
44 plot(time_axis,response);
45 title('Response of the system');

46 %% Dereverberation
47 figure(2);
48 subplot(212);
49 yd = filter(g,1,y);
50 time_axis = 0:1/fs_x:(length(yd)-1)/fs_x;
51 plot(time_axis,yd);
52 title('Dereverberated signal')
53 xlabel('Time [s]')
54 subplot(211);
55 plot(time_axis,y);
56 title('Original signal')
57 xlabel('Time [s]')
58

```

The function `pinv` computes the pseudoinverse and introduces also the regularisation even though it uses another type of regularisation.

What we obtain is

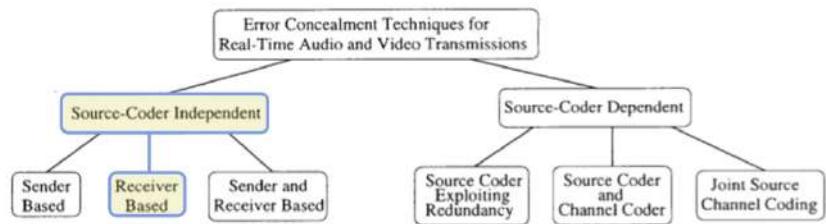


The difference is visible especially in the tails! The reverberant tails, in the second one, are more suppressed. Nowadays, we prefer to use other techniques that are more effectively.

## Local degradations

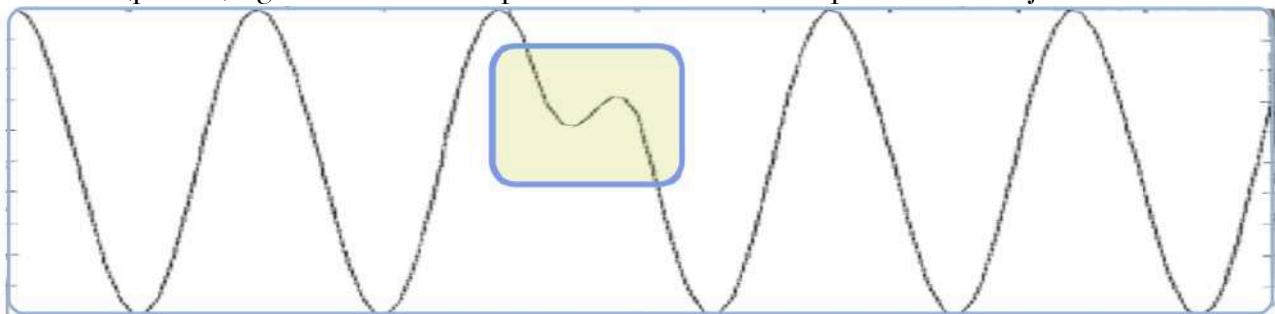
Now it is time to move towards local degradation. One of the typical problems of local degradation is the loss of one package during data transmission. The error comes from the fact that some information has been lost. There are two main classes of techniques to fix the problem:

- Source-Coder Independent:  
the source has not any redundant information that can be used to recover the package that have been lost.
- Source-Coder Dependent:  
the opposite.



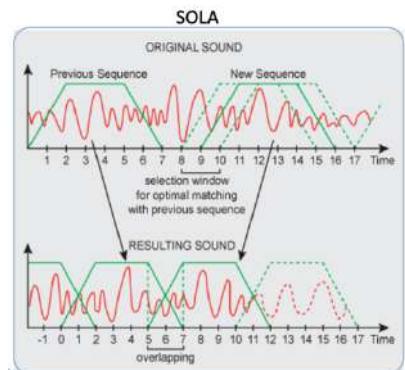
We will analyse a source-coder independent technique called receiver based.

Having lost the packet at  $M$ , the simplest solution is to replicate the packet at the sample  $M - 1$ . To do that, we use the overlap-and-add technique. Unfortunately, the overlap-and-add technique is not suitable for sinusoidal (pitched) signal. As we know, it produces artefacts in correspondence of the junction.

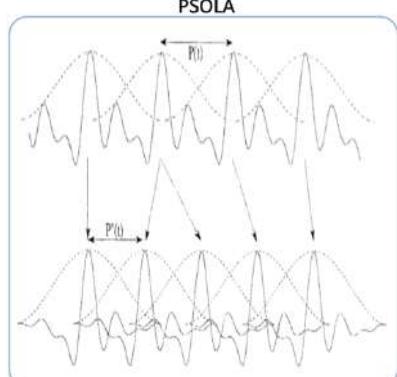


There are three main techniques used to overcome the problem of splicing artefacts introduced by OLA. Here are listed:

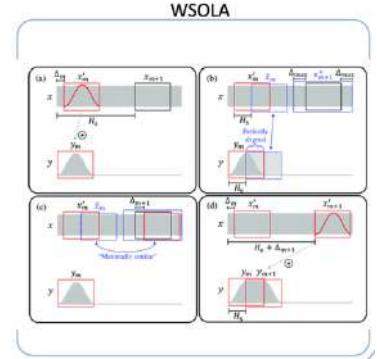
- Synchronised OLA (SOLA). Splices are repositioned in such a way as to correlate with the already formed portion.



- Pitch-Synchronous OLA (PSOLA). Even a small change in the formant location or bandwidth can considerably alter the naturalness of the modified speech. PSOLA has the advantage of not modifying the formants (unlike SOLA or Pitch Scaling). It splices the signal in a synchronous way with respect to the spikes. So, we window the signal by considering the centre of the window in correspondence of the spikes.



- Waveform Similarity OLA (WSOLA). Used in adaptive playout (when we want to modify the rate speed e.i. make the message on WhatsApp faster) is done by taking a segment known as template and searching back/forth for a best-match. When it is found, the best-match is mixed with the template. Finally, a longer/shorter segment is created by taking the mix and the rest of the segment from the best-match to the end of the packet.

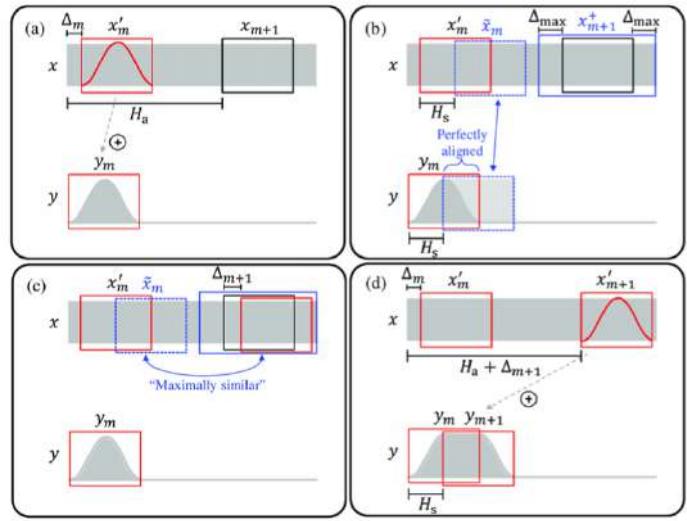


So, how do we proceed, for example, with a WSOLA?

We find a portion of the signal that is maximally similar to the one that was already in the outputs signal.

Step by step:

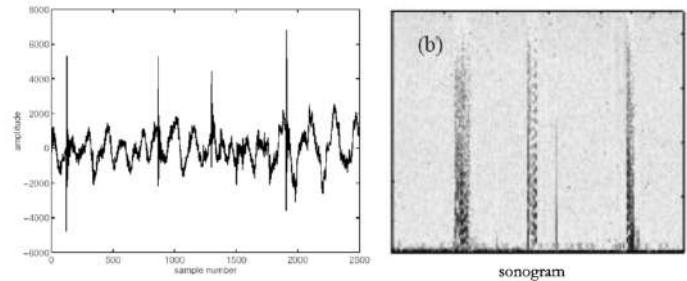
- Is given an input audio signal  $x$  with the adjusted analysis frame  $x_m$ . The frame was already windowed and copied to the output signal  $y$ ;
- Retrieval of a frame from the extended frame region  $x + m + 1$  (solid blue box) that is as similar as possible to the natural progression  $x_m$  (dashed blue box) of the adjusted analysis frame  $x_m$ ;
- Same
- The adjusted analysis frame  $x'_{m+1}$  is windowed and copied to the output signal  $y$ .



## Local degradation in the context of historical restauration

Historical recordings are often affected by defects of finite duration occurring at random locations and only affecting the signal in a localized fashion. They are perceived as tiny tick noises, which can occur in any recording medium including modern digital sources. More specifically, scratch and crackle noise are associated with most analogical disc recordings. Poor quality 78rpm record might typically exhibit around 2000 clicks per second with durations ranging from less than  $20 \mu s$  up to  $4 ms$  in extreme cases. In most cases at least 90% of the samples remains undegraded, therefore it is reasonable to hope that a convincing restoration can be achieved. The typical causes of such a degradation can be found in specks of dirt or dust adhering to the grooves of a disc, in the granularity in the material used for pressing and in damages such as surface scratches. By looking at the waveform on the right, we can identify the clicks in the spikes that are spread all along the track.

We need to find a technique that allows us to spot them algorithmically and solve them.



- Additive model: acceptable for most surface defects in recording media including small scratches, dust and dirt.
- Replacement model: appropriate for very large scratches and breakages, which completely obliterate any underlying signal information. Such defects usually excite long-term resonances in mechanical playback systems and must be treated differently.

We will now go through the first technique considering the additive model.

$$x(n) = s(n) + i(n)v(n)$$

Where:

$s(n)$  is the underlying audio signal

$v(n)$  is the corrupting noise process

$$i(n) = \begin{cases} 1, & \text{where the degradation is present} \\ 0, & \text{otherwise} \end{cases}$$

This kind of signals are characterised by contiguous bursts of corrupted samples starting at random positions and of random duration (1 to 200 samples at 44.1 kHz sampling rates). Furthermore, there is a strong dependence between successive samples of the switching process  $i(n)$ . As the switching process is, by definition, a binary one, noise cannot be assumed as impulsive. The fact that  $i(n)$  is a process and not a single spike, makes more difficult to conduct the detection. It is in fact more difficult to treat clusters of impulses than single impulses as the effects of adjacent impulses can cancel each other in detection space (missed detection) or add constructively to give the impression of more impulses (false detection).

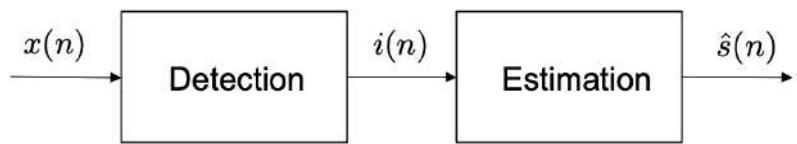
Another aspect that characterises clicks is a very variable amplitude. Largest click amplitudes will be well above the largest signal amplitudes; smallest audible defects can be more than 40 dB below the local signal level. This leads to some difficulties:

- Large amplitude defects tend to bias any parameter estimation and threshold determination procedures leaving smaller defects undetected.
- Threshold selection for some detection schemes becomes a difficult problem.

Let us see now how to approach the problem.

With respect to the global degradation, here instead we have the preliminary step of detecting the errors in the signals. The process is therefore composed by two steps:

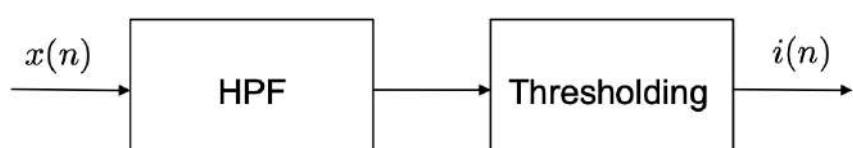
1. Detection: where  $i(n)$  is estimated so where it is decided which samples are degraded.
2. Signal estimation: where the underlying audio data is reconstructed where corruption is present. As we said, in the absence of information on the underlying audio data, the reconstruction is done from uncorrupted neighbouring samples.



A very first naïve attempt to detection is to magnify them over the signal through high pass filtering (they are very rapid, so they have lot of high frequencies (wide band)) and then apply thresholding. This solution has some advantages:

- It is simple to implement
- There are no unknown system parameters

Anyway, there is also a disadvantage: the detection relies on frequency separation between signal and clicks, which is rarely true...In fact, it exhibits problems when signal is high-pass or when clicks are band-limited (remember the degree of variability of clicks!)



And for the estimation?

The simplest solution is to interpolate the spectrogram of the samples of the window that are neighbouring the corrupted one. Another proposal is to use the median filter. It is a particular type of “filter” (non linear) whose idea is to take the median of a sequence for example, you have the following sequence

$$x = \{2, 80, 6, 3, 5, 4, \dots\}$$

Then you replace the value missing or the “strange” with the middle value made with the previous and the next one. In this case,

$y[1] = \text{Median}\{2, 2, 80\} = 2$  (the first 2 has been added in order to always have three values. Very often the first number is repeated).

$$y[2] = \text{Median}\{2, 80, 6\} = 6$$

$$y[3] = \text{Median}\{80, 6, 4\} = 6$$

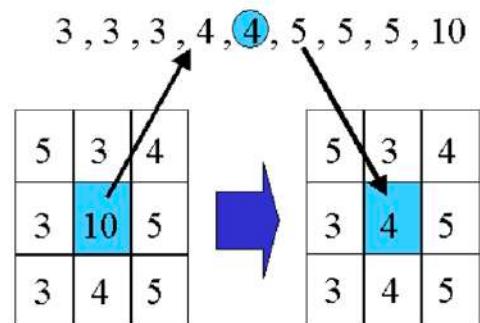
$$y[4] = \text{Median}\{6, 3, 5\} = 5$$

$$y[5] = \text{Median}\{3, 5, 4\} = 5$$

The restored signal will be  $y = \{2, 6, 6, 5, 5, \dots\}$

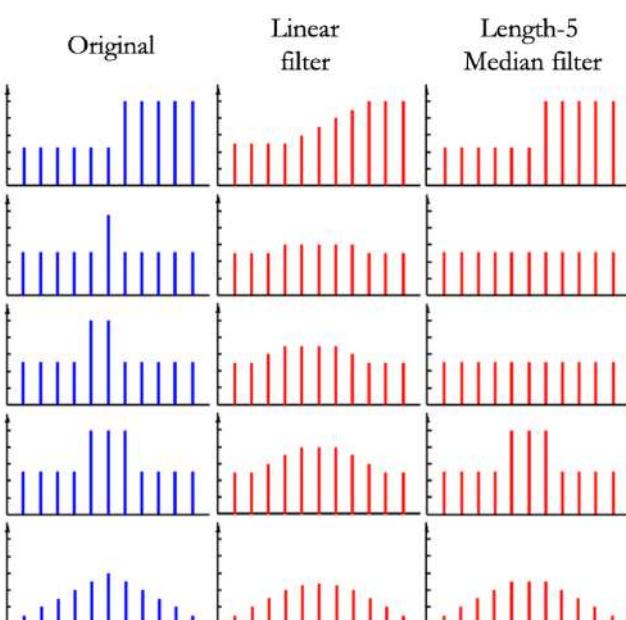
We now report, in summary, the most important property of such a filter.

- Loosely worded: it is not a filter! (not Linear Time-Invariant). The nonlinearity is due to sorting.
- It preserves step-like functions: useful when considering images (for the edges that pass from a colour to black).
- Removes isolated errors: a filter of length  $2N + 1$  can remove “outlier” sequences of up to  $N$  samples.
- It is suitable only for very short chunks of missing data (a few samples). Signal degradation soon becomes unacceptable. Furthermore, local details get lost!



On the right is reported a generalisation for 2d data (used for images).

In the image below we can see how changes the effect of a linear filter and of a median filter on a same signal.



The difference between a linear filter and the median filter lies in the fact that the median filter preserves the “discontinuities” (e.g. step function) and, at the same time, it deletes all the samples that are non-coherent with the others.

Let us see now a very stupid example on MATLAB about the using of the median filter.

```

15 I = floor(N*rand(num_clicks,1));
16 max_length = 10;
17 L = floor(max_length*rand(size(I)));
18 for j = 1:length(I)
19     end_index = min(I(j)+L(j)-1,length(s));
20     s(I(j):end_index) = max(min(0.5*randn(end_index-I(j)+1,1),1),-1);
21     %s(I(j):end_index) = ones(end_index-I(j)+1,1);
22 end
23 %soundsc(s,fs);

25 %% Plot the signal spectrogram
26 [S,f,t] = stft(s,fs,'FrequencyRange','onesided');
27 figure(1);
28 imagesc(t,flipud(f),abs(flipud(S)));
29 xlabel('Time[s]');
30 ylabel('Frequency [Hz]');

32 %% Design the filter for click detection. Cutoff: 1500 Hz, as observed fr
33 Wn = 4500 / (fs/2); % Cutoff frequency on normalized frequency scale
34 O = 10;
35 bhi = fir1(O,Wn,'high');
36 sd = filter(bhi,1,s); → He applies an
37 %% Click detection
38 mean_s = mean(sd);
39 std_s = std(sd);
40 thr = mean_s + 5*std_s;
41 K = find(abs(s) > thr);

42 figure(2);
43 subplot(312);
44 time_axis = 0:1/fs:(length(sd)-1)/fs;
45 plot(time_axis,abs(sd));
46 hold on;
47 plot(time_axis(K),sd(K),'or');

49 %% Median "filtering"
50 filter_length = 21;

51 sr = s;
52 for j = 1:length(K)
53     begin_index = max(K(j)-floor(filter_length/2),1);
54     | end_index = min(K(j)+floor(filter_length/2),length(s));
55     | sr(K(j)) = median(s(begin_index:end_index));
56 end → The low pass
57
58
59 %% Additional step: low pass filtering ← removes a bit
60 Wn = 2500 / (fs/2); % Cutoff frequency on normalized frequency scale
61 O = 10;
62 bhi = fir1(O,Wn,'low');
63 sr = filter(bhi,1,sr);
64 soundsc(sr,fs);

```

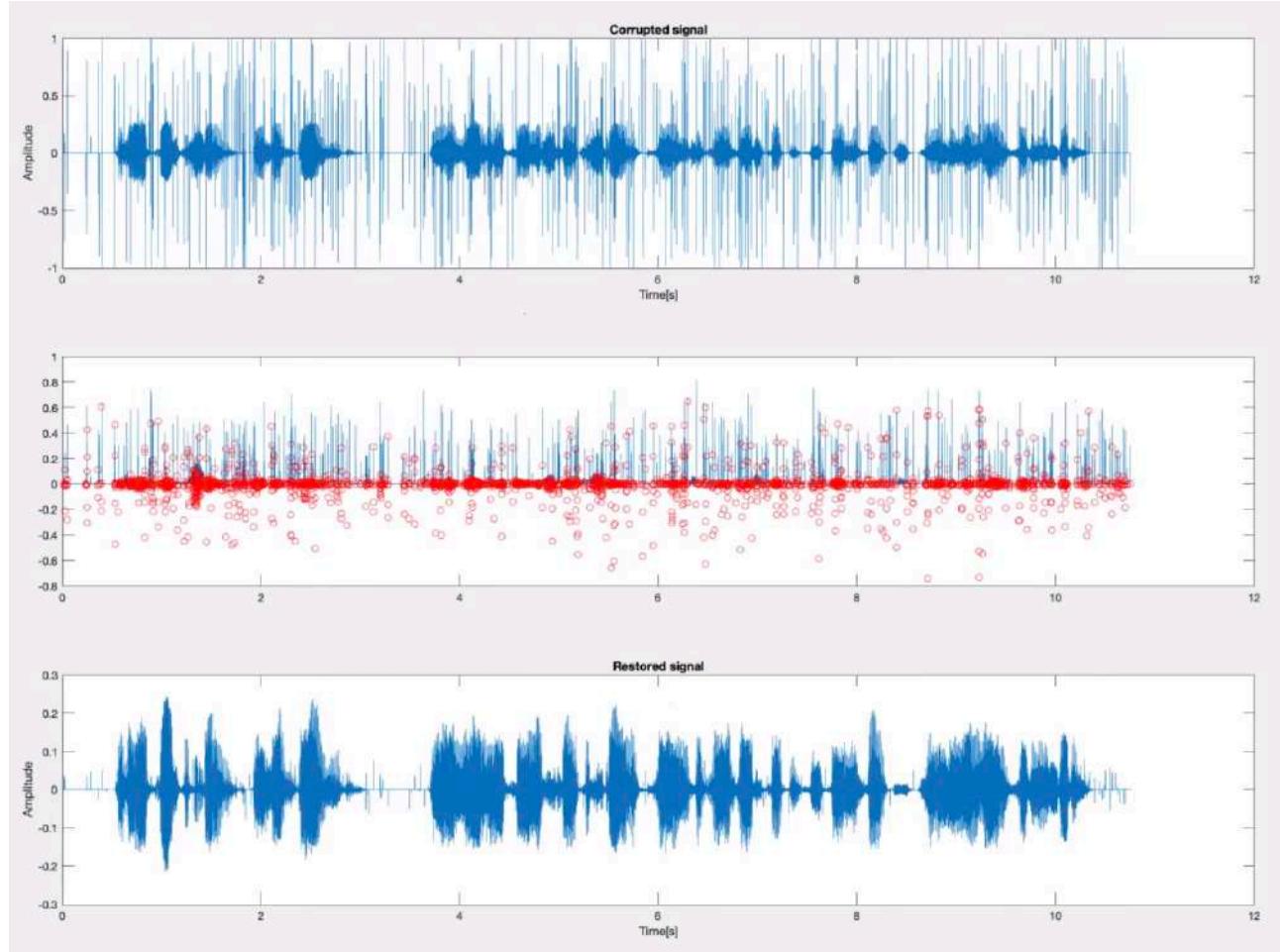
```

65 %% Plot the signals
66 time_axis = 0:1/fs:(length(s)-1)/fs;
67 subplot(311);
68 plot(time_axis,s);
69 xlabel('Time[s]');
70 ylabel('Amplitude');
71 title('Corrupted signal');
72 subplot(313);
73 plot(time_axis,sr);
74 xlabel('Time[s]');
75 ylabel('Amplitude');
76 title('Restored signal');
77
78
79
80

```

Command Window

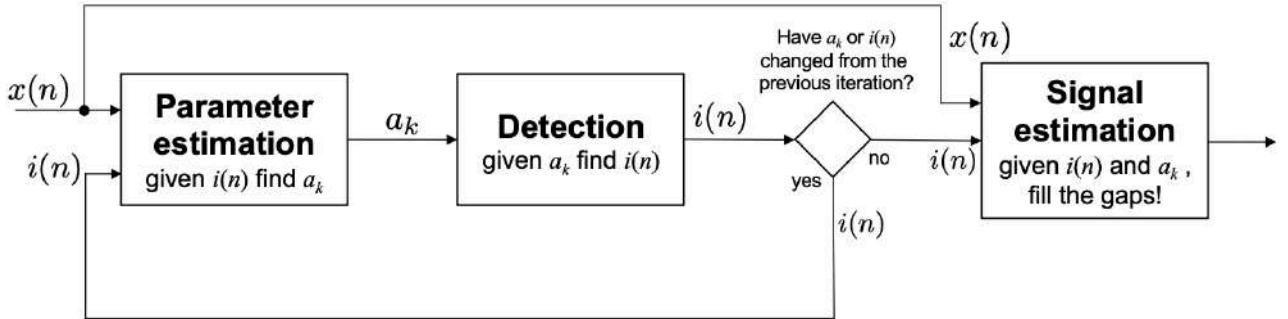
Let us see the result.



As we can notice the corrupted signal is full of clicks. The red dots are there for the detection. As we can see, with this quick and dirty algorithm, we can still achieve a satisfactory result.

## A General Scheme for Detection and Estimation

Let us now see a finer solution. Clicks have very variable magnitude and very variable duration. So, it is not a good idea to use a fixed scheme to delete them. It is better to use a scheme in which the two blocks (detection and estimation) are somehow interconnected with an autoregressive model.



### Click detection

Let us analyse the blocks. Audio data  $s(n)$  is assumed to be drawn from a short-term stationary AR process

$$s(n) = \sum_{k=1}^P a_k s(n-k) + e(n) \xrightarrow{a_k} \text{Detection given } a_k \text{ find } i(n) \rightarrow i(n)$$

We assume (for the time being) that the AR model parameters  $a_k$  and excitation variance  $\sigma_e^2$  are known. Let us switch to Frequency domain.

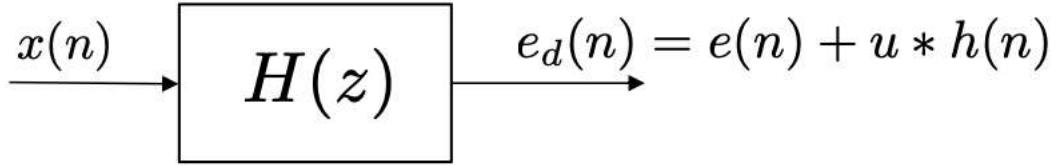
$$s(n) = \sum_{k=1}^P a_k s(n-k) + e(n) \xrightarrow{z} S(z) = S(z) \sum_{k=1}^P a_k z^{-k} + E(z)$$

$$H(z) = \frac{E(z)}{S(z)} = 1 - \sum_{k=1}^P a_k z^{-k}$$

What we obtained is the error prediction filter. Let us try to feed this filter with the corrupted data  $x(n)$ . What we get is the error signal

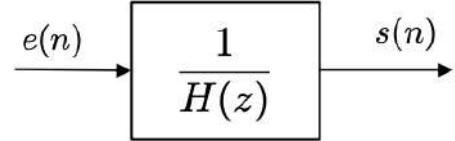
$$\begin{aligned} e_d(n) &= x(n) - \sum_{k=1}^P a_k x(n-k) \\ &\quad \stackrel{\substack{\text{Click disturbed model} \\ x(n)=s(n)+i(n)v(n)}}{=} \\ &= [s(n) + i(n)v(n)] - \sum_{k=1}^P a_k [s(n-k) + i(n-k)v(n-k)] = \\ &= s(n) - \underbrace{\sum_{k=1}^P a_k s(n-k)}_{e(n)} + i(n)v(n) - \sum_{k=1}^P a_k i(n-k)v(n-k) = \\ &= e(n) + i(n)v(n) - \sum_{k=1}^P a_k i(n-k)v(n-k) \end{aligned}$$

Therefore, we have



Where  $h(n)$  is the impulse response of the prediction error filter, and  $u(n) = i(n)v(n)$  is the localised noise. The response of the prediction error filter to the corrupted signal is basically the sole noise component  $e(n)$ . The filter, in fact, kills off the predictable component of the musical signal, while leaving the noise  $v(n)$  nearly unchanged.

Recalling that  $s(n) = \sum_{k=1}^P a_k s(n-k) + e(n)$ , therefore



This means that the variance of the prediction error can be derived from the variance of the uncorrupted signal  $s(n)$  as follows:

$$P_s(e^{j\omega}) = P_e(e^{j\omega}) \left| \frac{1}{H(e^{j\omega})} \right|^2$$

By integrating from  $-\pi$  to  $\pi$  we get that

$$\underbrace{\int_{-\pi}^{\pi} P_s(e^{j\omega}) d\omega}_{2\pi\sigma_s^2} = \int_{-\pi}^{\pi} P_e(e^{j\omega}) \left| \frac{1}{H(e^{j\omega})} \right|^2 d\omega = \sigma_e^2 \int_{-\pi}^{\pi} \left| \frac{1}{H(e^{j\omega})} \right|^2 d\omega$$

**Reminder:**

When a stochastic signal is a zero-mean signal, the variance is equal to the statistical power.

Which means that

$$\sigma_e^2 = \frac{2\pi\sigma_s^2}{\int_{-\pi}^{\pi} \left| \frac{1}{H(e^{j\omega})} \right|^2 d\omega}$$

This expression can be used to quantify the reduction in power of the audio signal, which turns out to be in the order of 40dB or more for highly correlated audio.

Along that, the expression

$$e_d(n) = e(n) + i(n)v(n) - \sum_{k=1}^P a_k i(n-k)v(n-k)$$

tells us that, if a click has the form of a single impulse, it triggers a full impulse response of the prediction error filter (weighted by the impulse's magnitude). This means that a relevant amplification is achieved for uncorrelated localised noise-like signals.

Click amplification is achieved at the expense of temporal localization (the effect spreads over  $P$  samples). This is a little drawback of this technique that, after the processing, it makes difficult to detect the actual allocation of the peaks. The energy is now spread on a few tens of samples. We gain in terms of detectability of the threshold, but we lose the temporal localisation.

Click detection is performed through thresholding of  $e_d(n)$ .

Threshold is determined using the magnitude of the peaks and the order of the model. The system has to be set to find the right trade-off between false positives and false negatives.

We summarise everything with a pseudo-code algorithm

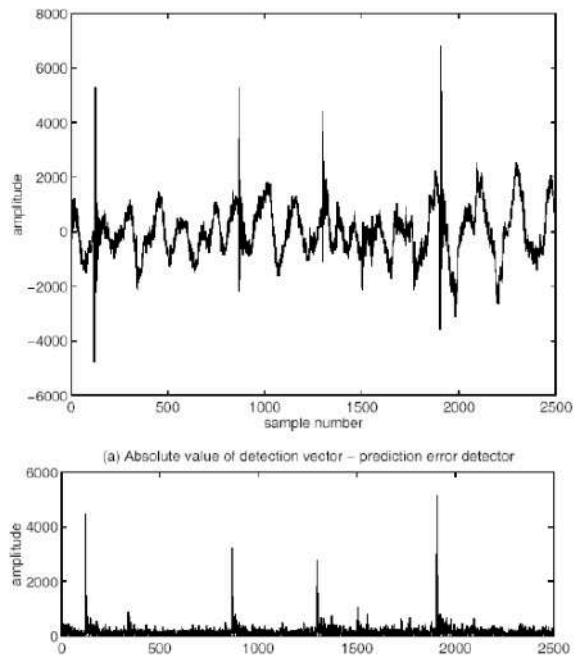
For  $n = 1$  to  $N$

    Compute prediction error

$$e(n) = x(n) - \sum_{i=1}^P a_i x(n-i)$$

    If  $|e(n)| > k\sigma_e$  then  $i(n) = 1$ , else  $i(n) = 0$

End



Time for another MATLAB example

```

1  %% Sound Analysis, Synthesis and Processing
2  % Click detection through AR filtering
3  % Fabio Antonacci, February 2024
4  clear;
5  close all;
6  %% Creates the uncorrupted signal
7  [s,fs] = audioread("a_Front.wav");
8  N = length(s);
9  %% Estimate the LPC filter coefficients of the signal and keeps them for
10 %% future use
11 N_lpc = 10;
12 h = lpc(s,N_lpc);
13
14 %% Creates the mask for corrupting the signal and applies the corruption
15 sigma_n = 0.3;
16 num_clicks = 15;
17 I = floor(N*rand(num_clicks,1));
18 max_length = 10;
19 L = floor(max_length*rand(size(I)));
20 for j = 1:length(I)
21     end_index = min(I(j)+L(j)-1,length(s));
22     s(I(j):end_index) = s(I(j):end_index)+sigma_n*randn(end_index-I(j)+1);
23 end
24 %% Performs filtering of the signal through the known prediction filter
25 e = filter(h,1,s);
26
27 [click_pk, click_locs] = findpeaks(e,"MinPeakHeight",0.1*max(abs(e)), "M");
28 %% Plots the signals
29 subplot(212);
30 time_axis = 0:1/fs:(length(s)-1)/fs;
31 plot(time_axis,e);

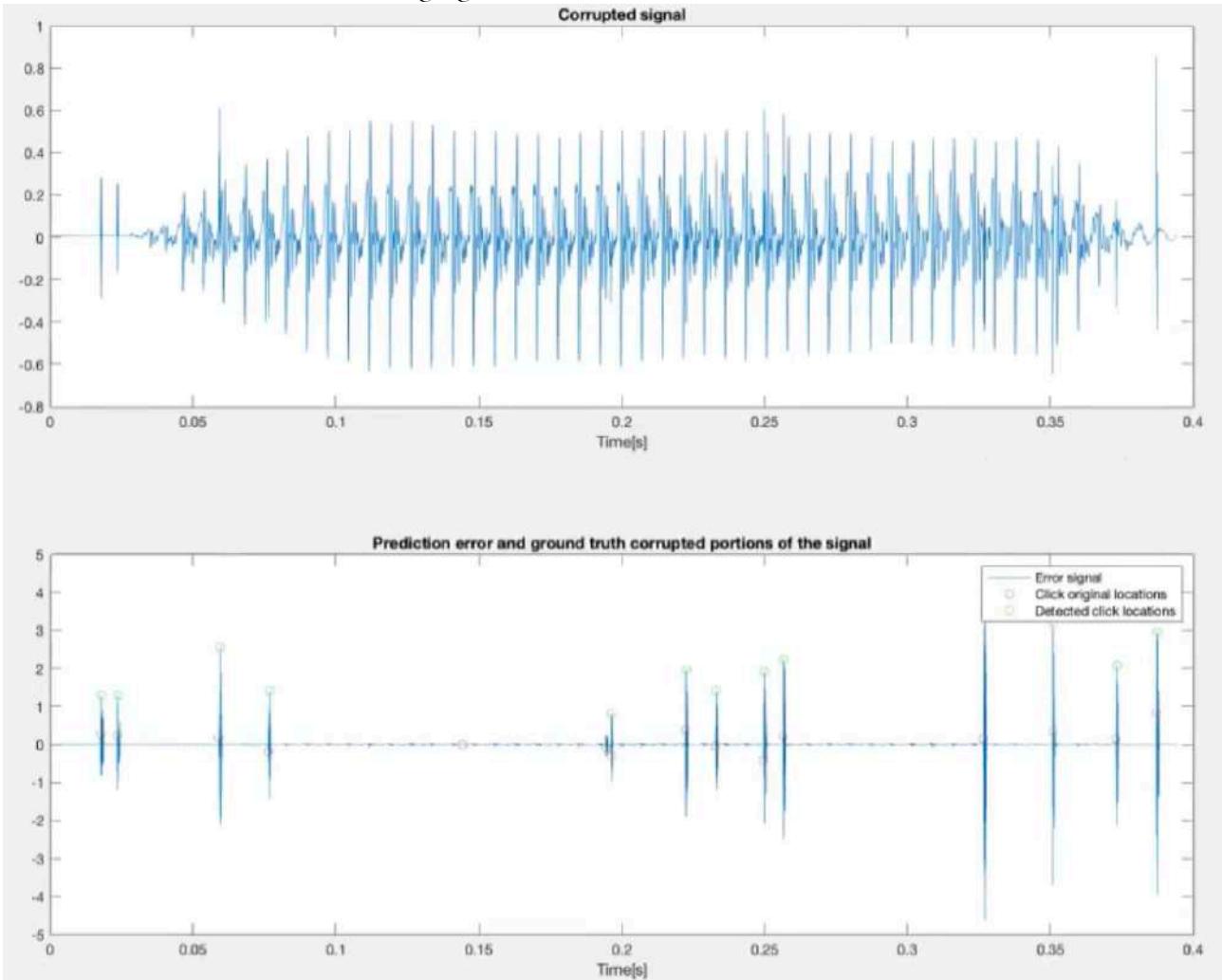
```

```

32 hold on;
33 plot(time_axis(I),e(I),'o');
34 plot(click_locs/fs,click_pk,'og')
35 xlabel('Time[s]');
36 title("Prediction error and ground truth corrupted portions of the signal")
37 legend('Error signal','Click original locations','Detected click locations')
38 subplot(211);
39 plot(time_axis,s);
40 xlabel('Time[s]');
41 title("Corrupted signal");

```

As a result we obtain the two following signals.



The green circles identify the clicks.

### Power Estimation

Let us now focus on the problem of modelling parameter estimation.

We start by highlighting some useful facts and some hypothesis.

- The model is linear in the parameters (simplifies the estimation)
- The Autoregressive model is correct for capturing the dynamics of the audio signal as the prediction error (innovation) is a white process (uncorrelated and identically distributed samples) and, for that reason, is Gaussian (which implies that uncorrelation implies independency)



Let us consider a random process  $\{x(n)\}$  drawn from an order-P Auto-Regressive (AR) filter driven by a white noise (innovation process)  $\{e(n)\}$

$$x(n) = \sum_{k=1}^P a_k x(n-k) + e(n)$$

Let  $p_e(e(n))$  be the Probability Density Function (PDF) of the innovation process. Then the probability of the next sample of the process given the previous P can be written as

$$P[x(n)|x(n-1), x(n-2), \dots, x(n-P)] = p_e\left(\underbrace{x(n) - \sum_{k=1}^P a_k x(n-k)}_{e(n)}\right)$$

As samples of  $e(n)$  are independent and identically distributed (i.i.d.), we can write

$$P[x(P+1), x(P+2), \dots, x(N)|x(1), x(2), \dots, x(P)] = \prod_{n=P+1}^N p_e\left(x(n) - \sum_{k=1}^P a_k x(n-k)\right)$$

Let us re-write everything in vectorial fashion

$$\underline{x} = \begin{pmatrix} x(1) \\ \vdots \\ x(N) \end{pmatrix} = \begin{pmatrix} \bar{x} \\ \hat{x} \end{pmatrix} \quad \text{where } \bar{x} = \begin{pmatrix} x(1) \\ \vdots \\ x(P) \end{pmatrix} \text{ and } \hat{x} = \begin{pmatrix} x(P+1) \\ \vdots \\ x(N) \end{pmatrix}$$

Then

$$x(n) = \sum_{k=1}^P a_k x(n-k) + e(n)$$

can be rewritten as

$$\begin{aligned} \hat{x} &= \begin{pmatrix} x(P+1) \\ \vdots \\ x(N) \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^P a_k x(P+1-k) \\ \vdots \\ \sum_{k=1}^P a_k x(N-k) \end{pmatrix} + \begin{pmatrix} e(P+1) \\ \vdots \\ e(N) \end{pmatrix} = \\ &= \underbrace{\begin{pmatrix} x(P) & x(P-1) & \cdots & x(2) & x(1) \\ x(P+1) & x(P) & \cdots & x(3) & x(2) \\ \vdots & \vdots & & \vdots & \vdots \\ x(N-1) & x(N-2) & \cdots & x(N-P+1) & x(N-P) \end{pmatrix}}_{G_x} \begin{pmatrix} a_1 \\ \vdots \\ a_P \end{pmatrix} + \begin{pmatrix} e(P+1) \\ \vdots \\ e(N) \end{pmatrix} \end{aligned}$$

The general linear model can therefore be written as

$$\hat{\underline{x}} = \underline{G}_{\underline{x}} \underline{a} + \underline{e}$$

This means that the conditional probability

$$P[x(P+1), x(P+2), \dots, x(N) | x(1), x(2), \dots, x(P)] = \prod_{n=P+1}^N p_e \left( x(n) - \sum_{k=1}^P a_k x(n-k) \right)$$

Can be rewritten as

$$P[\hat{\underline{x}}, \underline{x}, \underline{a}] = p_e \left( \hat{\underline{x}} - \underline{G}_{\underline{x}} \underline{a} \right)$$

And in the case of zero-mean Gaussian excitation of variance  $\sigma_e^2$  we can write

$$p_e \left( \hat{\underline{x}} - \underline{G}_{\underline{x}} \underline{a} \right) = \left( \frac{1}{2\pi\sigma_e^2} \right)^{\frac{N-P}{2}} e^{-\frac{1}{2\sigma_e^2} (\hat{\underline{x}} - \underline{G}_{\underline{x}} \underline{a})^T (\hat{\underline{x}} - \underline{G}_{\underline{x}} \underline{a})}$$

Let us now apply these considerations to our problem: we want to estimate the AR model parameters  $\underline{a}$  from the observed data  $\underline{x}$ . The Maximum Likelihood (ML) estimator treats the parameters as unknown constants about which we incorporate no prior information. The observed data  $\underline{x}$  is, however, considered random and we can often then obtain the PDF for  $\underline{x}$  when the value of  $\underline{a}$  is known. This PDF is termed the likelihood  $L(\underline{x}; \underline{a})$ , which is defined as

$$L(\underline{x}; \underline{a}) = P[\hat{\underline{x}}, \underline{x}, \underline{a}] = p_e \left( \hat{\underline{x}} - \underline{G}_{\underline{x}} \underline{a} \right)$$

The ML estimate for  $\underline{a}$  is then that value of  $\underline{a}$  that maximises the likelihood for given observation  $\underline{s}$ :

$$\underline{a}^{ML} = \arg \max_{\underline{a}} \{ L(\underline{x}; \underline{a}) \}$$

The rationale behind this is that the ML solution corresponds to the parameter vector which would have generated the observed data  $\underline{x}$  with highest probability. Usually, it is computed by maximizing the log-likelihood. In our case

$$\begin{aligned} \underline{a}^{ML} &= \arg \max_{\underline{a}} \{ L(\underline{x}; \underline{a}) \} = \arg \max_{\underline{a}} \left\{ \log \left( p_e \left( \hat{\underline{x}} - \underline{G}_{\underline{x}} \underline{a} \right) \right) \right\} = \arg \max_{\underline{a}} \left\{ -\frac{1}{2\sigma_e^2} \underline{e}^T \underline{e} \right\} = \\ &= \arg \min_{\underline{a}} \left\{ \underbrace{\underline{e}^T \underline{e}}_{E} \right\} \end{aligned}$$

Which can be found by setting

$$\frac{\partial E}{\partial \underline{a}} = 2\underline{e}^T \frac{\partial \underline{e}}{\partial \underline{a}} = 2 \left( \hat{\underline{x}} - \underline{G}_{\underline{x}} \underline{a} \right)^T \frac{\partial \left( \hat{\underline{x}} - \underline{G}_{\underline{x}} \underline{a} \right)}{\partial \underline{a}} = 2 \left( \hat{\underline{x}} - \underline{G}_{\underline{x}} \underline{a} \right)^T \underline{G}_{\underline{x}} = 0$$

This can be solved with respect to the parameter vector  $\underline{a}$  as

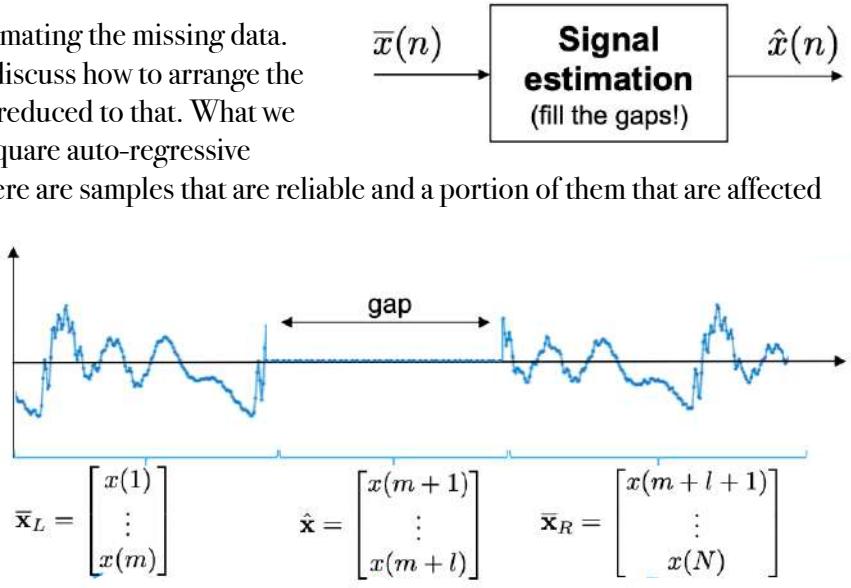
$$\underline{a}^{ML} = \left( \underline{\underline{G_x}}^T \underline{\underline{G_x}} \right)^{-1} \underline{\underline{G_x}}^T \hat{\underline{x}}$$

### Signal estimation

We now focus on the problem of estimating the missing data. Before doing that, it is important to discuss how to arrange the data because everything here can be reduced to that. What we are going to perform is called least-square auto-regressive

(LSAR) interpolation. In a signal, there are samples that are reliable and a portion of them that are affected by the clicks. Once this last portion is removed, a gap remains. Simply speaking, what we have is:

- Known samples on the left represented by  $\bar{\underline{x}}_L$
- Known samples on the right represented by  $\bar{\underline{x}}_R$
- Unknown samples (gap) represented by  $\hat{\underline{x}}$



We will have, therefore, two vectors:

$$\underline{x} = \begin{pmatrix} \bar{\underline{x}}_L \\ \hat{\underline{x}} \\ \bar{\underline{x}}_R \end{pmatrix} \quad \bar{\underline{x}} = \begin{pmatrix} \bar{\underline{x}}_L \\ \bar{\underline{x}}_R \end{pmatrix}$$

Let us rearrange the matrices:

$$\underline{x} = \begin{pmatrix} \bar{\underline{x}}_L \\ \hat{\underline{x}} \\ \bar{\underline{x}}_R \end{pmatrix}_{[N \times 1]} = \underbrace{\begin{pmatrix} 0_{m \times l} \\ I_{l \times l} \\ 0_{(N-m-l) \times l} \end{pmatrix}_{[N \times l]}}_U \hat{\underline{x}}_{[l \times l]} + \underbrace{\begin{pmatrix} I_{m \times m} & 0_{m \times (N-m-l)} \\ 0_{l \times m} & 0_{l \times (N-m-l)} \\ 0_{(N-m-l) \times m} & I_{(N-m-l) \times (N-m-l)} \end{pmatrix}_{[N \times (N-l)]}}_K \underbrace{\begin{pmatrix} \bar{\underline{x}}_L \\ \bar{\underline{x}}_R \\ \hat{\underline{x}} \end{pmatrix}}_{[(N-l) \times l]}}$$

From which we get

$$\underline{x} = \underline{\underline{U}} \hat{\underline{x}} + \underline{\underline{K}} \bar{\underline{x}}$$

Where:

- $\underline{\underline{U}}$  and  $\underline{\underline{K}}$  are the rearrangement matrices
- $\hat{\underline{x}}$  contains the unknown components
- $\bar{\underline{x}}$  contains the known components

We notice that

$$\hat{x} = \underline{G}_x \underline{a} + \underline{e} \rightarrow \underline{e} = \hat{x} - \underline{G}_x \underline{a}$$

By explicating the matrixes, we get

$$\begin{pmatrix} e(P+1) \\ \vdots \\ e(N) \end{pmatrix} = \begin{pmatrix} x(P+1) \\ \vdots \\ x(N) \end{pmatrix} - \begin{pmatrix} x(P) & \cdots & x(1) \\ \vdots & \ddots & \vdots \\ x(N-1) & \cdots & x(N-P) \end{pmatrix} \begin{pmatrix} a_1 \\ \vdots \\ a_P \end{pmatrix}$$

Which can be rewritten as

$$\underline{e} = \underline{\underline{A}} \underline{x}$$

Linear in  $\underline{x}$ . Let us make it explicit

$$\begin{pmatrix} e(P+1) \\ \vdots \\ e(N) \end{pmatrix} = \begin{pmatrix} -a_P & -a_{P-1} & \cdots & -a_1 & 1 & 0 & \cdots & 0 & \cdots & 0 & 0 \\ 0 & -a_P & \cdots & -a_2 & -a_1 & 1 & \cdots & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & -a_P & \cdots & -a_1 & 1 \end{pmatrix} \begin{pmatrix} x(1) \\ x(2) \\ \vdots \\ x(N) \end{pmatrix}$$

The conditional likelihood for white Gaussian excitation is then rewritten as:

$$P(\hat{x} | \bar{x}, \underline{a}) = \left( \frac{1}{2\pi\sigma_e^2} \right)^{\frac{N-P}{2}} e^{-\frac{1}{2\sigma_e^2} \underline{x}^T \underline{\underline{A}}^T \underline{\underline{A}} \underline{x}}$$

Let us now go deeper with LSAR. In order to express the unknown samples as a function of the known ones, we need to define an objective function to optimize (i.e., a cost function to minimize). We do so by defining an estimation error, and then by minimizing its energy (least squares method).

The cost function that we choose is the energy of the prediction error over the whole data block:

$$E = \underline{e}^T \underline{e} \quad \text{where } \underline{e} = \underline{\underline{A}} \underline{x}$$

We recall, in fact, that the block of data samples  $\underline{x}$  is assumed to be drawn from an AR process with coefficients (parameters)  $a_k$ . We have also already shown that the excitation vector  $\underline{e}$  can be written as a linear function of the whole block of data samples  $\underline{x}$ . We can therefore write:

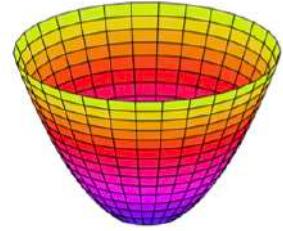
$$\underline{e} = \underline{\underline{A}} \underline{x} = \underline{\underline{A}} \left( \underbrace{\underline{U}}_{\underline{\underline{A}}} \hat{\underline{x}} + \underbrace{\underline{K}}_{\underline{\underline{A}}} \bar{\underline{x}} \right) = \underbrace{\underline{\underline{A}} \underline{U}}_{\underline{\underline{A}}} \hat{\underline{x}} + \underbrace{\underline{\underline{A}} \underline{K}}_{\underline{\underline{A}}} \bar{\underline{x}} \Rightarrow \underline{e} = \hat{\underline{\underline{A}}} \hat{\underline{x}} + \bar{\underline{\underline{A}}} \bar{\underline{x}}$$

obtaining a columnwise partition of  $\underline{\underline{A}}$  corresponding to unknown and known samples, respectively.

The LS interpolator works by minimizing the energy E of the prediction error.

$$\hat{x}_{LS} = \arg \min_{\hat{\underline{x}}} \{E\} = \arg \min_{\hat{\underline{x}}} \{\underline{e}^T \underline{e}\} = \arg \min_{\hat{\underline{x}}} \{\underline{x}^T \underline{\underline{A}}^T \underline{\underline{A}} \underline{x}\}$$

As the prediction error is a linear function of the data, the energy  $E$  is a paraboloidal function of the data, therefore it has a single global minimum, which can be found by setting its gradient to zero.



Therefore:

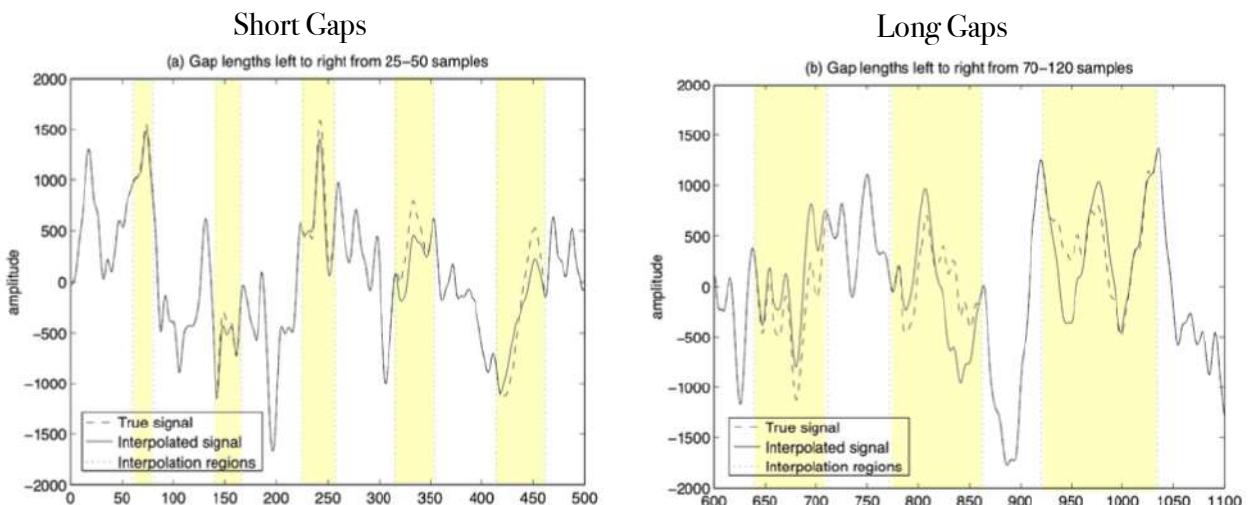
$$\begin{aligned} \frac{\partial E}{\partial \hat{x}} &= \frac{\partial}{\partial \hat{x}} \underline{e}^T \underline{e} = 2 \underline{e}^T \frac{\partial \underline{e}}{\partial \hat{x}} = 2 (\underline{\hat{A}} \hat{x} + \underline{\bar{A}} \bar{x})^T \frac{\partial (\underline{\hat{A}} \hat{x} + \underline{\bar{A}} \bar{x})}{\partial \hat{x}} = 2 (\underline{\hat{A}} \hat{x} + \underline{\bar{A}} \bar{x})^T \underline{\hat{A}} = 0 \\ \Rightarrow 2 (\underline{\hat{A}} \hat{x} + \underline{\bar{A}} \bar{x})^T \underline{\hat{A}} &= 0 \Rightarrow (\underline{\hat{A}} \hat{x} + \underline{\bar{A}} \bar{x})^T \underline{\hat{A}} = 0 \Rightarrow \hat{x}^T \underline{\hat{A}}^T \underline{\hat{A}} + \bar{x}^T \underline{\bar{A}}^T \underline{\hat{A}} = 0 \Rightarrow \\ \Rightarrow \hat{x}^T \underline{\hat{A}}^T \underline{\hat{A}} &= -\bar{x}^T \underline{\bar{A}}^T \underline{\hat{A}} \Rightarrow (\hat{x}^T \underline{\hat{A}}^T \underline{\hat{A}})^T = (-\bar{x}^T \underline{\bar{A}}^T \underline{\hat{A}})^T \Rightarrow \underline{\hat{A}}^T \underline{\hat{A}} \hat{x} = -\underline{\bar{A}}^T \underline{\bar{A}} \bar{x} \Rightarrow \\ \Rightarrow \hat{x} &= \underline{\hat{x}}_{LS} = -(\underline{\hat{A}}^T \underline{\hat{A}})^{-1} \underline{\hat{A}}^T \underline{\bar{A}} \bar{x} \end{aligned}$$

The method involves the solution of a set of  $l$  linear equations,  $l$  being the number of missing samples in the block. When there are at least  $P$  known samples either side of a single contiguous burst of missing samples the LSAR estimate can be efficiently implemented:

- Levinson-Durbin recursion (Toeplitz system of equations)
- computational load:  $O(l^2)$  multiplies and additions

### Performance

This algorithm works well in most cases but less well for signals that do not fit modelling assumptions (periodic pulse-driven voiced speech). We obtain dull results with longer gaps. Situation improves by increasing order, Loss of noisy components can be perceived.



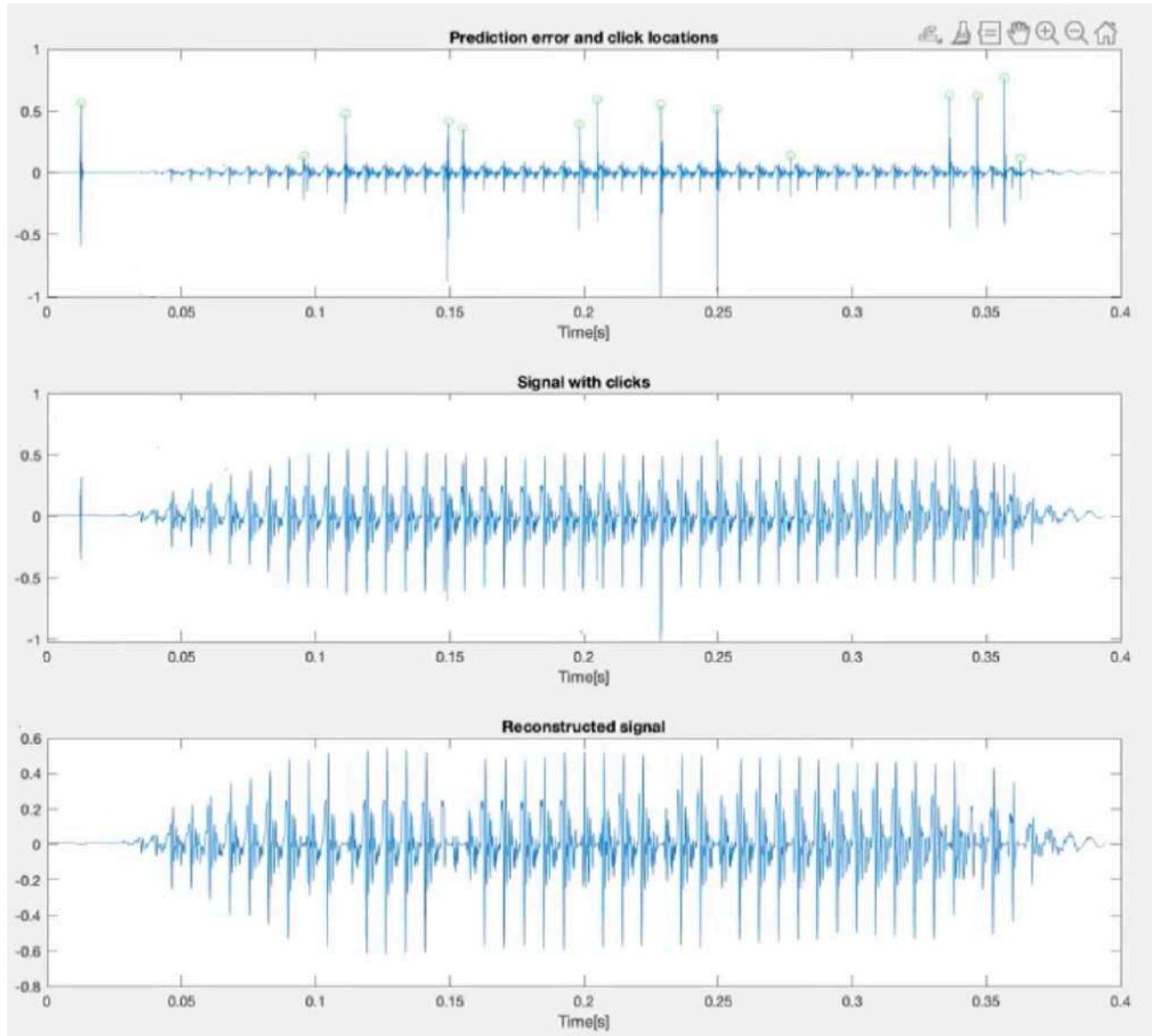
Once again, MATLAB example

```

1 %% Sound Analysis, Synthesis and Processing
2 % Gaps filling through LSAR approach
3 % Fabio Antonacci, February 2024
4
5 %% Performs preliminary click detection
6 click_detection_ar;
7 %% Defines algorithm parameters
8 click_window = max_length+2*(N_lpc-1); % Maximum duration of the click
9 window_extension = 3; % Left and right windows extensions
10 left_window_length = click_window+window_extension; % Length of the le
11 right_window_length = click_window+window_extension;
12
13 %% Click filling
14 sr = s; %Initialization of the restored signal
15 % For every detected click
16 for i = 1:length(click_locs)
17     % Find the beginning and end of the click window
18     start_index = max([click_locs(i)-click_window+1,1]);
19     end_index = min([click_locs(i)+click_window,length(s)]);
20     xh = s(start_index:end_index);
21     click_length = length(xh);
22     % Defines the left and right windows
23     start_left = max([start_index - left_window_length,1]);
24     xhl = s(start_left:start_index - 1);
25     end_right = min([end_index+right_window_length-1,length(s)]);
26     xhr = s(end_index:end_right);
27     % Performs the filling only if both left and right windows are present
28     if ~isempty(xhl) && ~isempty(xhr)
29         U=[zeros(length(xhl),click_length);eye(click_length,click_length)
30         K = [[eye(length(xhl)),zeros(length(xhl),length(xhr))];[zeros(cl
31         A = convmtx(h,size(K,1)-length(h)+1);
32         Ah = A*U;
33         Ao = A*K;
34         s_ls = -inv(Ah'*Ah)*Ah'*Ao*[xhl;xhr];
35         sr(start_index:end_index) = s_ls;
36     end
37 end
38 %% Plot the results of the filling
39 figure;
40 subplot(311);
41 plot(time_axis,e);
42 hold on;
43 plot(click_locs/fs,click_pk,'og');
44 xlabel('Time[s]');
45 title('Prediction error and click locations');
46 subplot(312);
47 plot(time_axis,s);
48 xlabel('Time[s]');
49 title('Signal with clicks');
50 subplot(313);
51 plot(time_axis,sr);
52 xlabel('Time[s]');
53 title('Reconstructed signal');

```

Let us see the results



The reconstructed signal seems more corrupted. Why? Because we used an auto regressive model. This problem can be overcome by using the long term LPC. In this way, we use a larger observation window in order to estimate the current one. Inside this larger window, peaks can be found.

Anyway, if we hear the result, the quality is improved.

## Linear Predicting Coding

In today's lecture we will talk about LPC, an efficient way to represent signals. One of the signals that can be represented with this optimised-performance technique is the voice.

We know that the voice is produced by the vocal cord (source) and "shaped" by the mouth, the nose and the vocal tract (filter). LPC is a signal modelling that follows this kind of source filter approach, it becomes very useful to represent all the sounds that have a pitch and that are reshaped by a filter.

This model finds a wide application in phone conversation. It is also used to model some kind of instrument sounds for instance the one of a plucked string, but it cannot be used for other instruments. Other interesting application are the vocoder, the talking box and all the effects that reproduce the sound of a voice.

### Main idea:

A certain signal  $s(n)$  can be approximated using a linear combination of its past samples:

$$s(n) \approx a_1 s(n-1) + a_2 s(n-2) + \dots + a_p s(n-p) = \sum_{k=1}^p a_k s(n-k)$$

If we assume that our signal is stationary (reasonable assumption for speech ONLY IF we work in frame-by-frame fashion, so with short portions of speech sound), we can then exploit statistical signal processing to find the coefficients that best approximate our signal.

### How to achieve it:

In order to do it, we introduce the autoregressive model where a signal  $s(n)$  can be modelled

$$s(n) = \underbrace{\sum_{k=1}^p a_k s(n-k)}_{\text{Part that depends on the signal itself (Autoregressive)}} + G \underbrace{u(n)}_{\text{White noise}}$$

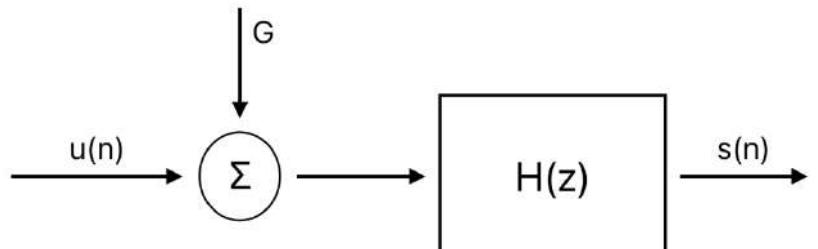
Where  $u(n) \sim \mathcal{N}(0, \sigma_u^2)$

We look for the transfer function by switching to Z domain:

$$S(z) = \sum_{k=1}^p a_k z^{-k} S(z) + G U(z) \rightarrow H(z) \triangleq \frac{S(z)}{G U(z)} = \frac{1}{1 - \sum_{k=1}^p a_k z^{-k}} = \frac{1}{A(z)}$$

From this expression we notice that the filter has surely a feedback loop (since it has poles). From that we derive a first "condition of existence": to assure that the filter is stable, all the poles must be inside the unit circle.

The model is in fact.



$H(z)$  is called Shaping filter. It is called in this way since, being an all-pole filter, has some peaks that reshape the signal put as input.

So, since we predict the signal, we need some sort of measure that tells us how good our prediction is. For this reason, we define the prediction error:

$$e(n) \triangleq s(n) - \hat{s}(n)$$

Where  $\hat{s}(n)$  is the prediction made by our LPC model  $\hat{s}(n) \triangleq \sum_{k=1}^p a_k s(n-k)$

In z-domain, the expression becomes:

$$\begin{aligned} E(z) &= S(z) - \hat{S}(z) = S(z) - \sum_{k=1}^p a_k S(z) z^{-k} = S(z) \underbrace{\left(1 - \sum_{k=1}^p a_k z^{-k}\right)}_{A(z)} = S(z) A(z) = \\ &= \frac{S(z)}{H(z)} \stackrel{H(z)=\frac{S(z)}{GU(z)}}{=} \frac{S(z)}{S(z)} GU(z) \end{aligned}$$

Why whitening filter? Because it takes  $S(Z)$  as input and make it white, in the sense that it removes all the peaks introduced by the filter obtaining, in this way, a flat spectrum: exactly the spectrum of white noise.

So, to fix some important definitions, let us recap them:

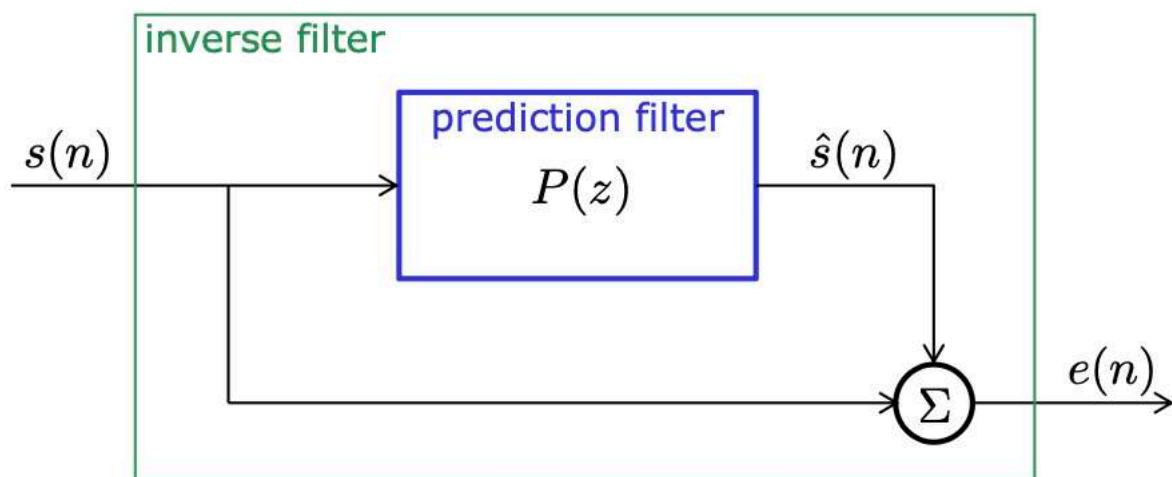
- $H(z)$  is the **shaping (or forward) filter**;
- $A(z)$  is **whitening (or inverse) filter**;

We now define also another filter, the **prediction filter**

$$P(z) = \sum_{k=1}^p a_k z^{-k}$$

That uses the previous sample to predict the new one.

The overall scheme is



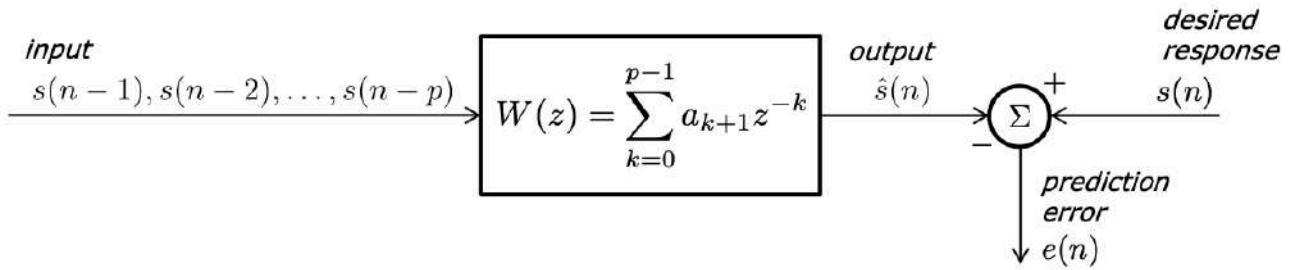
Where the prediction filter is included in the scheme of the whitening filter.

We assume to be able to predict a signal. All we have to do is to find the coefficient to define the predict filter that are the coefficient of  $P(z)$  able to minimise the MSE of the prediction error (function that is called **mean square prediction error**)

$$\min_{a_k} \{ |s(n) - \hat{s}(n)|^2 \} = \min_{a_k} \{ |e(n)|^2 \}$$

The problem can be set up as a Wiener Filtering problem, where the voice signal  $s(n)$  constitutes both the filter input and the desired response. Hence, there is a fundamental difference between LPC and Wiener filter. In the latter we want to find the coefficients to delete some components from the input signal to obtain a desired one, while in the former the input signal and the desired one are the same!

The input signal is a delayed version of the desired one



Keeping an eye on this difference, we can exploit the Wiener-Hopf equations to find the samples as we did for the Wiener filter

$$r(i) = E\{s(n)s(n-i)\}$$

So, we write the Wiener-Hopf equation.

$$\sum_{k=1}^p a_k r(i-k) = r(i) \quad i = 1, \dots, p$$

We know how to find the solution of this equation.

Since the order of the equation is  $p$ , we will need  $p$  coefficients that we collect in a vector. In order to find it, we write the equation in the vectorial fashion:

$$\underline{a} = \underline{\underline{R}}^{-1} \underline{r}$$

Where  $\underline{\underline{R}}$  is the autocorrelation matrix. Pay attention to the fact that we are using REAL values, so the lower triangle of the matrix is equal to the upper one (in the case of complex-value matrix, in the lower triangle we have the complex conjugate of the element we have in the upper triangle)

$$\underline{\underline{R}} = \begin{bmatrix} r(0) & r(1) & \cdots & r(M-1) \\ r(1) & r(0) & \cdots & r(M-2) \\ \vdots & \vdots & \ddots & \vdots \\ r(M-1) & r(M-2) & \cdots & r(0) \end{bmatrix}$$

$\underline{r}$  is the vector of the autocorrelation of  $s(n)$

$$\underline{r} = \begin{bmatrix} r(1) \\ \vdots \\ r(p) \end{bmatrix}$$

We start from 1 and not from 0.

And  $\underline{a}$  is the vector of filter coefficients:

$$\underline{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_p \end{bmatrix}$$

So, the right coefficients will be the ones that will minimise the cost function. We call the minimised cost function  $D_p$

$$D_p \triangleq J_{\min} = \sigma_s^2 - \underline{r}^T \underbrace{\underline{R}^{-1} \underline{r}}_{\underline{a}^T \underline{R}^{-1} \underline{a} = E\{s(n)s(n-0)\} = r(0)} = r(0) - \underline{r}^T \underline{a} = r(0) - \sum_{k=1}^p a_k r(k)$$

In this way, we replaced vectors with scalars.

We want to understand how the prediction is going. In order to do it, we define the prediction gain ( $G_p$ )

$$G_p \triangleq \frac{\sigma_s^2}{D_p}$$

The prediction Gain goes to infinity if  $D_p$  tends to zero. This result is reached in the ideal case, when we can perfectly predict the signal so, mathematically speaking, when  $s(n)$  is highly predictable.

On the other hand, if  $s(n)$  is unpredictable (for example if it is white noise), the autocorrelation goes to zero everywhere but in zero (since the autocorrelation evaluated in zero is the variance and if the variance is zero the signal is not a random signal). This means that  $D_p$  goes to  $\sigma_s^2$ . And the prediction gain to 1.

At the end we can affirm that, if we are around 1, it means that either we did not do a good job or that the model is not adapt for the signal. Otherwise, if it is very high, we achieved a good result.

### Statistical interpretation of the infinite memory linear prediction

So, if we have the coefficients  $a_k$  with which to predict  $\hat{s}(n)$  given its previous samples, we can write that

$$\hat{s}(n) = \sum_{k=1}^{\infty} a_k s(n-k)$$

We remind that the orthogonality principle for this case affirms that

$$E\{e_o(n)s(n-i)\} = 0, \quad i = 1, 2, \dots, \infty$$

What does it mean? That the optimal error is completely decorrelated from the samples of the signal  $s(n-i)$  with  $i \neq 0$ .

Assuming that we are in an optimal condition (so the orthogonality holds), let us see what happens to the error:

$$r_{e_o}(i) = E\{e_o(n)e_o(n-i)\} \stackrel{e_o(n-i)=s(n-i)-\hat{s}(n-i)}{=} E\{e_o(n)[s(n-i) - \hat{s}(n-i)]\} \stackrel{\hat{s}(n)=\sum_{k=1}^{\infty} a_k s(n-k)}{=} E\left\{e_o(n) \left[ s(n-i) - \sum_{k=1}^{\infty} a_k s(n-k-i) \right] \right\}$$

The sum goes up to  $\infty$  since we use the infinite memory LPC.

$$\begin{aligned}
&= \underbrace{E\{e_o(n)s(n-i)\}}_{=0} - E\left\{e_o(n) \sum_{k=1}^{\infty} a_k s(n-k-i)\right\} = \\
&= \begin{cases} 0 - 0 & \text{for } i \in 0, \dots, \infty \\ 0 - E\left\{e_o(n) \sum_{k=1}^{\infty} a_k s(n-k-0)\right\} & \text{for } i = 0 \\ 0 - 0 & \text{for } i \in 0, \dots, -\infty \end{cases} = \\
&= \begin{cases} 0 & \text{for } i \neq 0 \\ -E\{e_o(n)\hat{s}(n)\} & \text{for } i = 0 \end{cases}
\end{aligned}$$

The autocorrelation is even, so if it is 0 for  $i > 0$ , the same it is for  $i > 0$

For  $i = 0$ , the autocorrelation of  $e_o(n)$  will be its variance. In fact

$$r_{e_o}(i=0) = E\{e_o(n)e_o(n-0)\} = E\{e_o(n)e_o(n)\} \triangleq \sigma_{e_o}^2$$

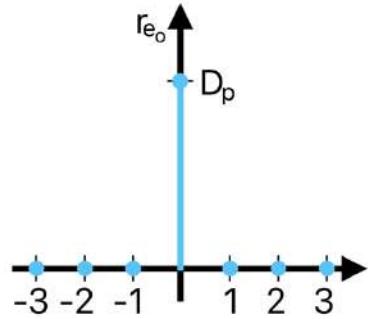
But also, since we are considering the optimal error

$$E\{e_o(n)e_o(n)\} = E\{|e_o(n)|^2\} = J_{min} = D_p$$

So,

$$r_{e_o}(0) = D_p \rightarrow r_{e_o}(i) = D_p \delta(i)$$

If we represent the auto correlation, what we obtain is a function with only one spike in  $i = 0$ , equals to  $D_p$ . This exactly the autocorrelation of a white noise.



White noise that should be exactly the output of the inverse filter (that for this reason is also called whitening filter). So, with the optimal coefficient,  $e_o(n)$  has this kind of autocorrelation.

At the same time, if we consider the inverse filter with optimal coefficients and ideal condition (so  $\infty$  number of coefficients) we can perfectly extract all the information that we can obtain from the past sample  $s(n)$  obtaining, as a result, everything that cannot be predicted (so everything that is random).

In this condition,  $s(n)$  is uncorrelated from  $e(n)$  and just by exploiting the prediction filter (whose coefficients are  $a_k$ ) we can obtain  $s(n)$  from it. This is very interesting in terms of coding because all the information we need to perform this operation are inside the prediction filter and we are also able to determinate the first order statistical index of the error (the variance).

### Recreate the signal from filter coefficients and white noise

We exploit this information to recreate the signal  $s(n)$ . We know that the inverse of a whitening filter is the shaping filter. This is what it is done with LPC in terms of coding:

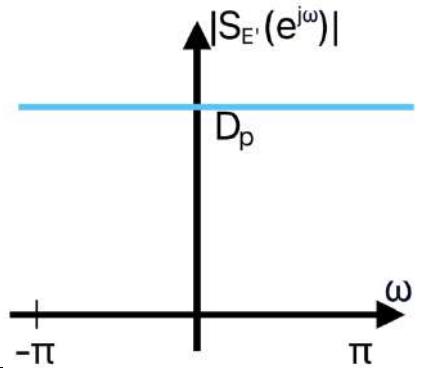
- we filter an LPC model finding the coefficients of the prediction filter
- we transmit only the filter coefficients and the information related to the prediction error (the variance  $\sigma_{e_o}^2$  that we found with the whitening filter) maximising the optimisation.
- Once we received the signal, we synthetise a white noise with the variance information we received from the transmission and we recreate the signal by mixing it with the sound filtered by the shaping filter (whose coefficients are evaluated from the one that have been transmitted).

So let us define an arbitrary white noise, we call  $e'(n) \sim \mathcal{N}(0, \sigma_{e'}^2)$ .

We decide to put  $\sigma_{e'}^2 = \sigma_{e_o}^2 = D_p$ .

We might know that the power spectrum of a white signal corresponds to a constant value equal to its autocorrelation.

$$|E'(e^{j\omega})|^2 = |E_o(e^{j\omega})|^2 = D_p$$



Reminder:

Why the power spectrum  $S(f)$  of a white signal is equal to its variance for all the values of  $f$ ?

$$\lim_{T \rightarrow \infty} \frac{1}{T} \left| X(f) \text{rect}\left(\frac{f}{T}\right) \right|^2 = \lim_{T \rightarrow \infty} \frac{1}{T} |X_T(f)|^2 \triangleq S(f) \stackrel{\substack{\text{Wiener-Hinčin's} \\ \text{Theorem}}}{=} \mathcal{F}\{r(\tau)\} \stackrel{\substack{\text{White noise}}}{=} \mathcal{F}\{r(0)\delta(\tau)\} = S(f) = \mathcal{F}\{r(\tau)\}$$

$$\stackrel{\substack{\text{r}(0)=\sigma_x^2}}{=} \mathcal{F}\{\sigma_x^2 \delta(\tau)\} = \sigma_x^2 \mathcal{F}\{\delta(\tau)\} = \sigma_x^2$$

So, thanks to the convolution theorem we can write that

$$|S'(e^{j\omega})|^2 = |H(e^{j\omega})|^2 |E'(e^{j\omega})|^2 = |H(e^{j\omega})|^2 D_p$$

And

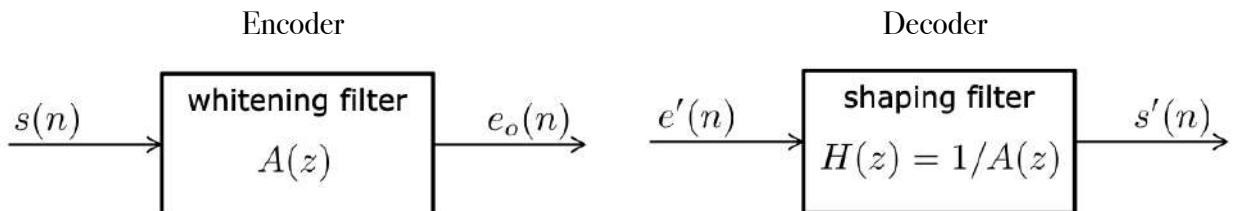
$$|S(e^{j\omega})|^2 = |H(e^{j\omega})|^2 |E(e^{j\omega})|^2 = |H(e^{j\omega})|^2 D_p$$

They have the same power spectrum even though the signal is different ( $s(n) \neq s'(n)$ ).

This means that if we find the optimal prediction filter, we can transmit  $A(z)$  together with the information of  $D_p$  and recreate a signal that will have different phase but same energy! Perfect to understand speech.

We know that there are different excitation signals suited for different type of speech. We choose the one that suits it better.

Let us draw the block diagram:



## Dealing with non-stationary signal

All we saw so far is very nice, but there are some problems:

- The coefficients must change in time since we do not want to transmit the same sound from the start to the finish -> we overcome this problem with an Overlap and Add approach.
- We do not have infinite memory as we took for granted in our discussion.
- The voice signal is not stationary! To overcome this limitation, we perform the LPC analysis over short segments of the signal, over which the signal is assumed as stationary.

For all these reasons, let us put a limit on the memory on M values.

We define

$$s_n(m) \triangleq s(n - m) \quad m = 0, \dots, M - 1$$

$$e_n(m) \triangleq e(n + m) \quad m = 0, 1, \dots, M + P - 1$$

Result of a convolution:  
length of the samples of  
 $s(n)$  length of the  
samples of the impulse  
response of the filter - 1.

So we look for the LPC parameters (time-varying) that minimise the short-time mean squared error:

$$\min_{\underline{a}_n} \varepsilon \quad \text{where } \varepsilon_n = \sum_{m=0}^{M+P-1} e_n^2(m) = \sum_{m=0}^{M+P-1} \left[ s_n(m) - \sum_{k=1}^p a_k(n) s_n(m-k) \right]^2$$

We already know how to find these coefficients: with the Wiener-Hopf equation! To find it, we need to estimate the samples of the auto-correlation function; in particular, we are interested in the short-time auto-correlation.

$$r_n(|i - k|) \triangleq \sum_{m=0}^{M-1-(i-k)} s_n(m) s_n(m+i-k) \quad \begin{array}{l} 1 \leq i \leq p \\ 0 \leq k \leq p \\ i \geq k \end{array}$$

Now that we know the auto-correlation function, we can write the Wiener-Hopf equations

$$\sum_{k=1}^p a_k(n) r_n(|i - k|) = r_n(i) \quad \text{with } i = 1, 2, \dots, p$$

We write them in matrix form:

$$\underline{a}_n = \underline{\underline{R}}_n^{-1} \underline{r}_n$$

with:

$$\underline{r}_n = \begin{bmatrix} r_n(1) \\ \vdots \\ r_n(p) \end{bmatrix} \quad \underline{a}_n = \begin{bmatrix} a_1(n) \\ \vdots \\ a_p(n) \end{bmatrix} \quad \underline{\underline{R}}_n = \begin{bmatrix} r(0) & r(1) & \cdots & r(p-1) \\ r(1) & r(0) & \cdots & r(p-2) \\ \vdots & \ddots & \ddots & \vdots \\ r(p-1) & r(p-2) & \cdots & r(0) \end{bmatrix}$$

Solving this for every time frame is very complex because the computation complexity grows cubically with the dimension of the matrix, so we use an iterative method called **Levinson-Durbin Recursion**. This algorithm exploits the Toeplitz structure of the auto-correlation matrix  $\underline{\underline{R}}_n$ . The idea is to apply a tapered window to the extracted frames (possibly with overlap) to attenuate edge effects.




---

*Tapered window*   *Finestra di Tukey*

Reminder:

A Toeplitz matrix or diagonal-constant matrix, named after Otto Toeplitz, is a matrix in which each descending diagonal from left to right is constant.

Example

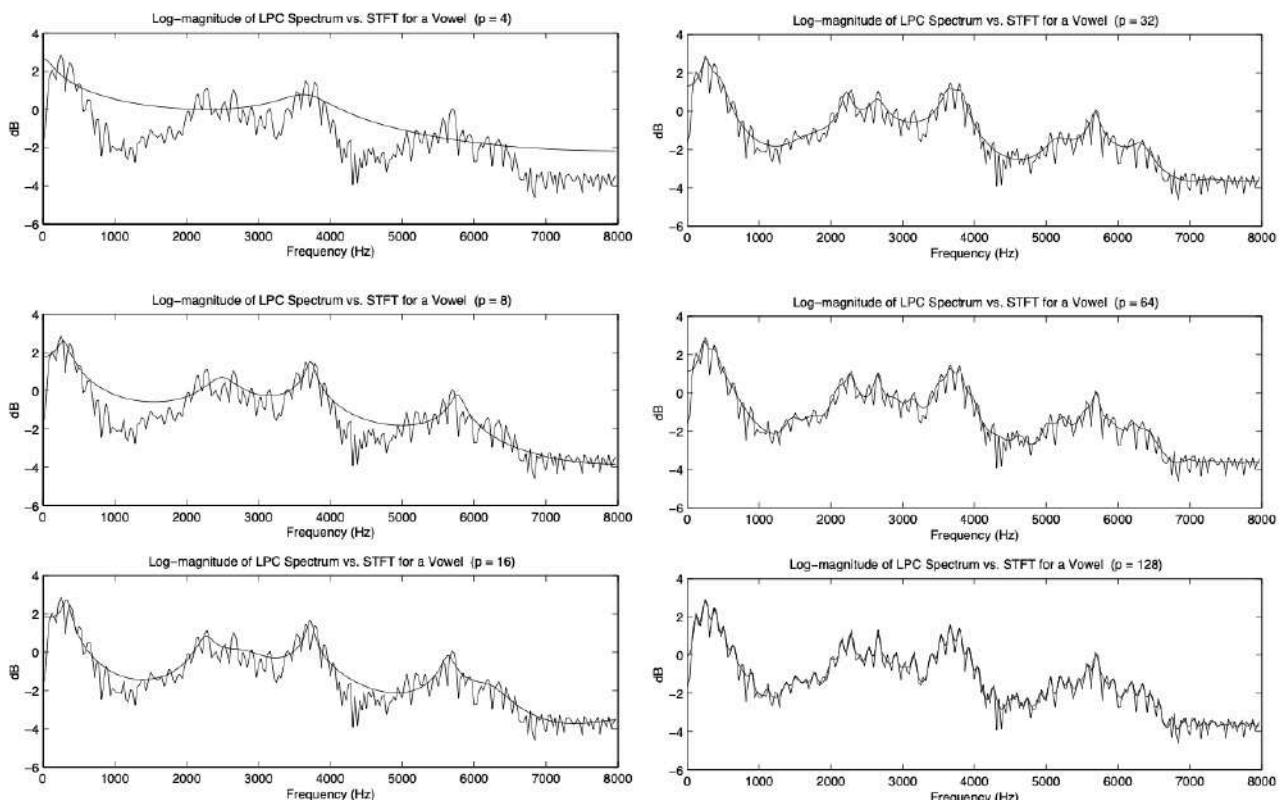
$$\begin{pmatrix} a & f & g & h & i \\ b & a & f & g & h \\ c & b & a & f & g \\ d & c & b & a & f \\ e & d & c & b & a \end{pmatrix}$$

We saw earlier that in the case of infinite memory LP the power spectrum of the signal can be exactly reconstructed from the shaping filter and the error variance. This implies that, as  $p \rightarrow \infty$ , we can approximate the power spectrum of the signal with arbitrarily small error using the all-pole shaping filter  $H_n(z)$ . Express this concept with maths, we have:

$$\lim_{p \rightarrow \infty} |\hat{S}_n(e^{j\omega})|^2 = |S_n(e^{j\omega})|^2$$

The greater is the number of coefficients, the better we can approximate the spectrum. This is why as the prediction order increases, the resulting mean-squared error  $\varepsilon_n$  monotonically decreases. This implies that as we increase the prediction order, the LPC power spectrum  $|\hat{S}_n(e^{j\omega})|^2$  will try to match the signal power spectrum more closely  $|S_n(e^{j\omega})|^2$ .

In the following picture, we can see exactly how increases the precision of  $|\hat{S}_n(e^{j\omega})|^2$  as  $p$  increases.



We notice that using a low number of coefficients, we match the greatest spikes in the spectrum.

So, we said that the short-time error is time limited to the interval  $[0, M + p - 1]$ , thus it can be expressed as

$$\varepsilon_n = \sum_{m=0}^{M+p-1} e_n^2(m) \underset{\substack{\text{The others remain} \\ \text{zero}}}{\underset{\omega}{\equiv}} \sum_{m=0}^{\infty} e_n^2(m)$$

By applying the Parseval's theorem

Reminder:

The Parseval's Theorem affirms that there is a formal equality in the integral of the product between one signal and the complex conjugate of another one in time domain and the sum of their coefficients in frequency domain and vice-versa.

$$\sum_n x[n]y^*[n] = \frac{1}{2\pi} \int_{[2\pi]} X(e^{j\omega})Y^*(e^{j\omega})d\omega$$

If  $x[n] = y[n]$  and  $x[n] \in \mathbb{R}$ , we have

$$\sum_n x[n]x^*[n] = \sum_n x^2[n] = \frac{1}{2\pi} \int_{[2\pi]} |X(e^{j\omega})|^2 d\omega$$

we can write that:

$$\begin{aligned} \varepsilon_n &= \sum_{m=0}^{M+p-1} e_n^2(m) \underset{\substack{\text{The others remain} \\ \text{zero}}}{\underset{\omega}{\equiv}} \sum_{m=0}^{\infty} e_n^2(m) \underset{\substack{\text{Parseval}}}{\underset{\omega}{\equiv}} \frac{1}{2\pi} \int_{-\pi}^{\pi} |E_n(e^{j\omega})|^2 d\omega = \\ &= \frac{1}{2\pi} \int_{-\pi}^{\pi} |S_n(e^{j\omega})|^2 |A_n(e^{j\omega})|^2 d\omega = \frac{1}{2\pi} \int_{-\pi}^{\pi} \frac{|S_n(e^{j\omega})|^2}{|H_n(e^{j\omega})|^2} d\omega \end{aligned}$$

Since the integrand is positive (ratio between two modulus) we conclude that,

$$\min_{a_n} \varepsilon_n \Leftrightarrow \min_{a_n} \frac{|S_n(e^{j\omega})|^2}{|H_n(e^{j\omega})|^2}, \quad \forall \omega$$

In order to find the coefficients that minimise the error, we have to find the coefficients that minimise the ratio.

But why if we use a low number of coefficients, we match the greatest things in the power spectrum? That is because with a low number of coefficients  $s(n)$  is very strong (it means that  $|S_n(e^{j\omega})|^2$  is very high), so the magnitude of  $S$  is greater than the magnitude of  $H$

$$|S_n(e^{j\omega})| > |H_n(e^{j\omega})|$$

So the ratio will be positive and the overall error will be high. In order to reduce the error, we try to match the frequencies for which the condition is true, and these are exactly the peaks! So we try to find the coefficients  $a_k$  that in the first instance approximate them.

On the other hand, in the valley of the frequency response, the value of the error is already low because the signal  $s(n)$  is not strong there. In fact, there

$$|S_n(e^{j\omega})| < |H_n(e^{j\omega})|$$

The thing is that the region in which  $|S_n(e^{j\omega})| > |H_n(e^{j\omega})|$  contributes more to the overall error rather than region where  $|S_n(e^{j\omega})| < |H_n(e^{j\omega})|$ . Since we have a low number of coefficients, we prefer to spend them to match the region in which the signal is more relevant. And that is why LPC can be used for tracking the power spectrum, because we will first match the component of the power spectrum that have more energy (so the one that are more relevant) and, by increasing the number of coefficients, we can also match the regions that are not important.

All of this just to say that we have to respect a trade-off between increasing and decreasing the number of coefficients:

- By increasing them we reduce the error making the two signals match but at the same time we have more coefficients to send and the communication is not optimised.
- By decreasing the number of coefficients, we pick only the important part of the signal but we lose the quality.

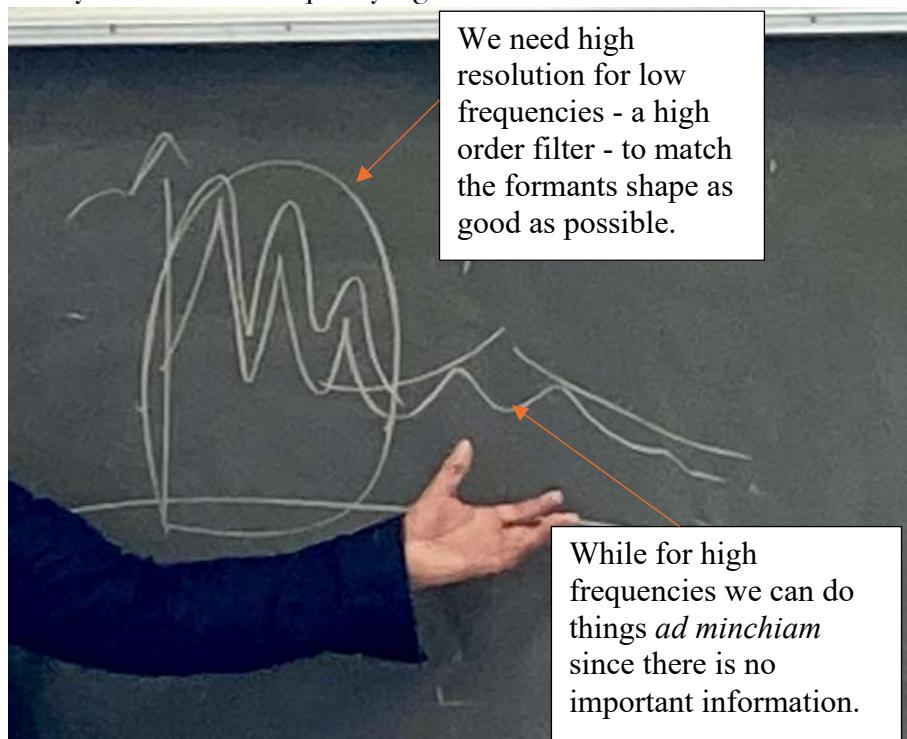
Which is the right number? We do not know, depends on the signal! There is a rule of thumb used in literature to tell us which should be the order of the filter.

Given the sampling frequency, the order  $p$  should be between

$$\frac{F_s}{1000} \leq p \leq \frac{F_s}{1000} + 4$$

In a normal situation, we use more or less 20 coefficients.

Sometimes we want more control on what we do: our hearing system follow a logarithmic behaviour according to our psychoacoustic capability. Hence, we are more interested in modelling low frequencies with higher accuracy with respect to higher frequency because the sensitive of human ear scales logarithmically with the frequency (we can discard the high frequency and focus on low ones). Usually, the formants in a speech signal have strong resonances at low frequencies but just very wide resonances at high frequency. It would be smart to have variable order with respect to the frequency. So we are interested in exactly match the low frequency region.



Usually, the signal is divided in frequency bands and the estimation of filter coefficients is made with different resolution for each of them. Typical values are

- 20 for low frequencies;
- <20 for high frequencies.

With this strategy we achieve an even more compact representation of our signal.

## Application of LPC:

### *Speech coding synthesis*

One of the most useful applications of LPC is speech-coding synthesis. We can exploit the fact that most of the information of voice filtering are compatible with LPC decomposition.

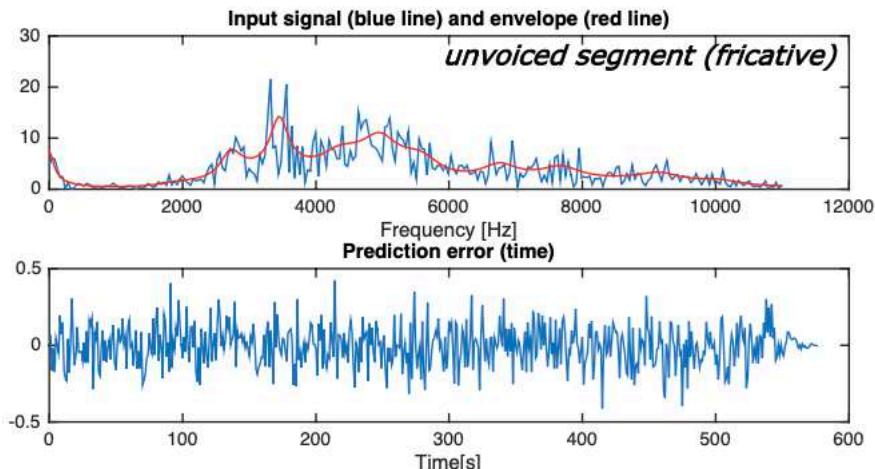
How can we transmit speech using LPC?

1. We divide the speech signal in a series of short windows in order to make the signal stationary inside them;
2. We retrieve the coefficients of the LPC filter. Following the rule of thumb, we need only 20 coefficients to reconstruct a message correctly: an optimal compression if we compare it to the amount of data we need for an entire speech signal.
3. We compute the variance of the noisy signal we get at the end of the decomposition.
4. We send all this information to the recipient.
5. We reconstruct the signal:
  - a. We will need just the variance of noise for unvoiced segments as source. The signal produced will be filtered by the LPC filter whose transfer function is the inverse of the one whose coefficients are the ones that has been sent.

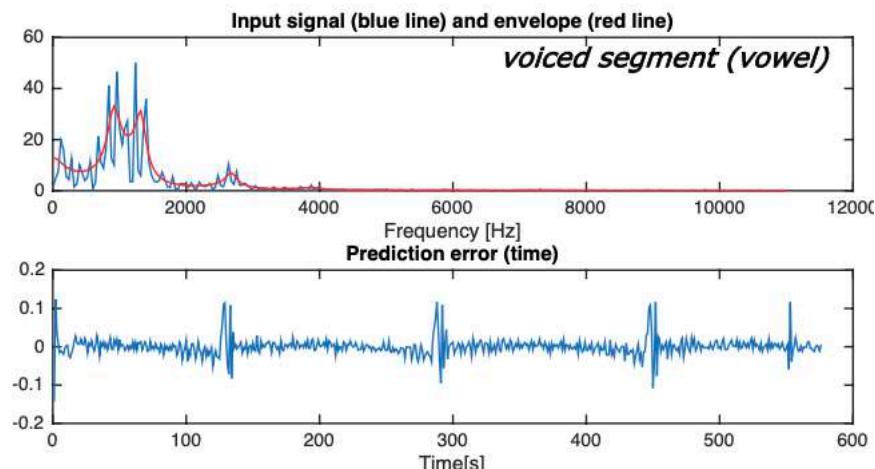
---

	<i>Recipient</i>	<i>Destinatario</i>
--	------------------	---------------------

---

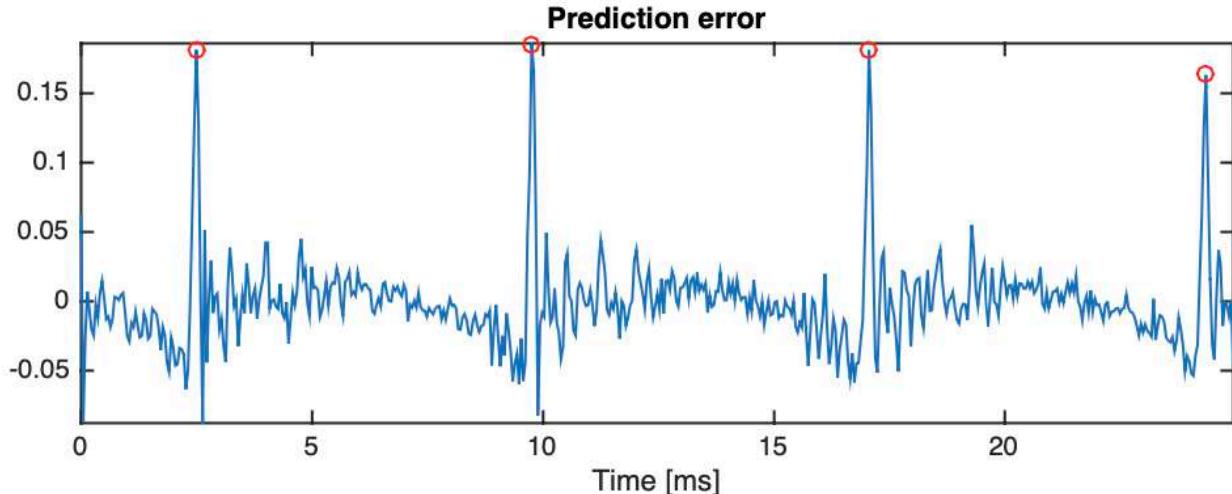


- b. On the other hand, for voiced signal we recreate the envelop by means of the coefficients but, instead of using a white noise source, we generate a train of pulses as source signal. To get the proper frequency we exploit the fact that, usually, the prediction error tends to present some spikes periodically. This periodicity is exactly the fundamental frequency of the voice.



### *Prediction pitch*

We can use LPC also for other applications, for example the prediction pitch. If the cropped portion of a voiced signal is large enough to see some periodicities in the prediction error, we know that the inverse of the period between them will be the fundamental frequency of the voiced signal we are analysing.



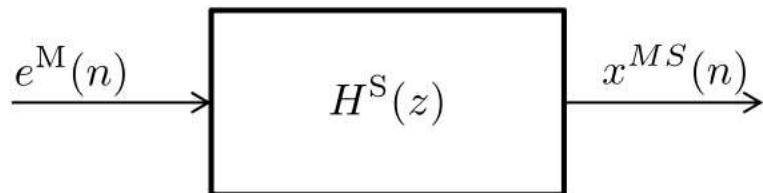
A more robust estimation can be obtained considering a long-term predictor. We just have to look at one period back in the past to have more information to better predict what it will come next.

### *Cross synthesis (or talking instruments)*

This is basically the digital version of the talking box effects. The analogical version consists basically in a sound that enters in the mouth with a tube and that is filtered by the vocal tract. This effect is widely used from Stevie Wonder, Daft Punk (for example Around the world). We can use LPC in order to recreate it. We feed a sound to the shaping filter  $H_n(z)$ , filter that has been obtained from LPC analysis of a speech segment. The musical signal input acts as a periodic excitation source  $e_n(m)$  to the shaping filter  $H_n(z)$ , and thus the output spectrum will possess vocal formant structure as well as the harmonic and textural qualities of the musical sound.

The methodology we have to follow is this one:

- For each frame, we perform LPC analysis on the musical signal  $x^M(n)$  and the speech signal  $x^S(n)$
- Then, we use the prediction error  $e^M(n)$  of the musical segment to feed the shaping filter  $H^S(z)$  of the speech segment.



## Time and Pitch Scaling

The goal of time and pitch scaling is the one to control temporal and pitch evolution of an audio signal. With time scaling we want to control the speed and the duration of the signal without altering the spectral components. Pitch scaling is its counterpart: we want to transpose the song without changing its speed.

Application:

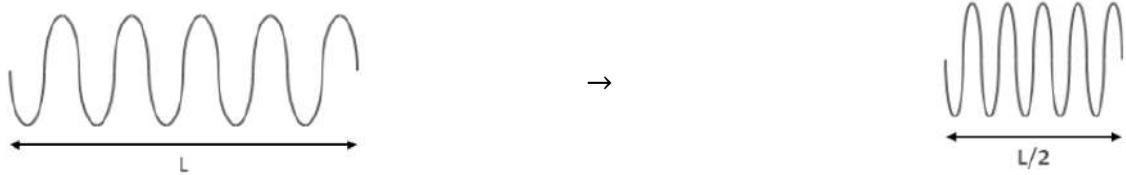
- Speed up a message on WhatsApp.
- Reading for the blind
- Computer interface
- Musical composition: in order to mix and match different materials (for example Dj Shadow).

The problem is clear, the applications are many. Which are the issues?

The problem lies in the fact that time and frequency domain are linked by definition. If you ever tried to slow a vinyl down, you noticed how, with speed, changes also pitch. That's due to the scaling theorem:

$$x(t) \xrightarrow{\mathcal{F}} X(f) \Rightarrow x(\alpha t) \xrightarrow{\mathcal{F}} \frac{1}{|\alpha|} X\left(\frac{f}{\alpha}\right)$$

The more we compress in time domain, the more we expand in frequency domain (for example when we subsample the signal). In order to have a practical sensation of what it happens, let us assume that we want to half the length of a sinusoidal signal. If we do so by subsampling of a factor 2, we obtain a signal with half duration but the frequency increases by a factor 2 as well!



We have to fool Fourier scaling theorem by outfoxing it! How can we formalise this problem?

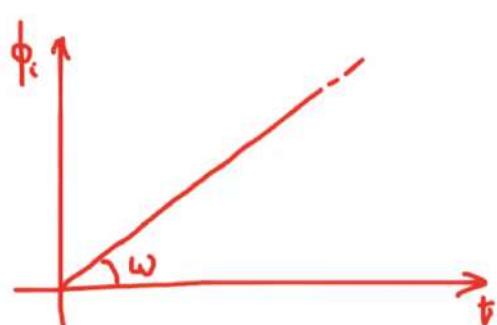
Since time and frequency of a general signal are not independent, we come up with a **parametric model** of audio signal. So, hereinafter we will consider a **sinusoidal model** assuming that each signal can be modelled with a sum of multiple sinusoids. So

Where:

$$x(t) = \sum_{i=1}^{I(t)} A_i(t) e^{j\phi_i(t)}$$

$I(t)$  is the number of sinusoids that compose the sound at time  $t$   
 $A_i(t)$  is the amplitude of the  $i$ -th sinusoid at time  $t$   
 $\phi_i(t) = \int_{-\infty}^t \omega_i(\tau) d\tau$  -> the phase is the integral of the frequency of the sinusoid

A sinusoid is a signal whose phase changes linearly in time whose slope corresponds to the frequency of the sinusoid. From the spectral point of view, by using the complex exponential we consider only the positive frequency spectrum. With a cosine, we consider also the negative one. Speaking of audio signal, with sinusoids and complex exponential we can talk about the same thing. We prefer to use the complex exponential for exploiting the analytical signal, but we can think it as sinusoid.



## Time Scaling

Our problem is that when we deal with complex number, we have amplitude and phase, but frequency is the variable on which we want to work. Our desired time-scaled signal is the following:

$$x'(t') = \sum_{i=1}^{I(T^{-1}(t'))} A_i(T^{-1}(t')) e^{j\phi'_i(t')} \quad \phi'_i(t') = \int_{-\infty}^{t'} \omega_i(T^{-1}(\tau)) d\tau$$

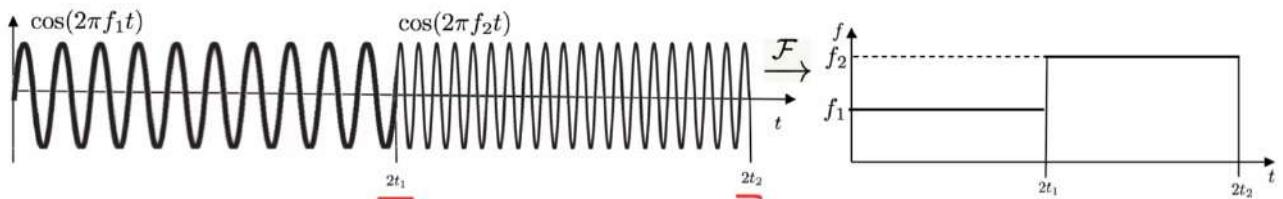
Where  $T^{-1}(t')$  is the **time warping function** that specifies a mapping between the time domain in the original signal  $t$  and the time domain modified signal  $t' = T(t)$ . So, basically, we can compute which is the new time instant  $t'$  of the scaled signal by applying this function to the instant  $t$  of the original signal.

If we read this equation with this in mind, it happens that the new signal at time  $t'$  will have as many sinusoids  $I()$  that it had at the time  $t$ , that is  $T^{-1}(t')$ . In fact, the sum goes from 1 to  $I(T^{-1}(t'))$ . Each one of those has the amplitude that had in the original signal at time  $t = T^{-1}(t')$  so  $A_i(T^{-1}(t'))$ .

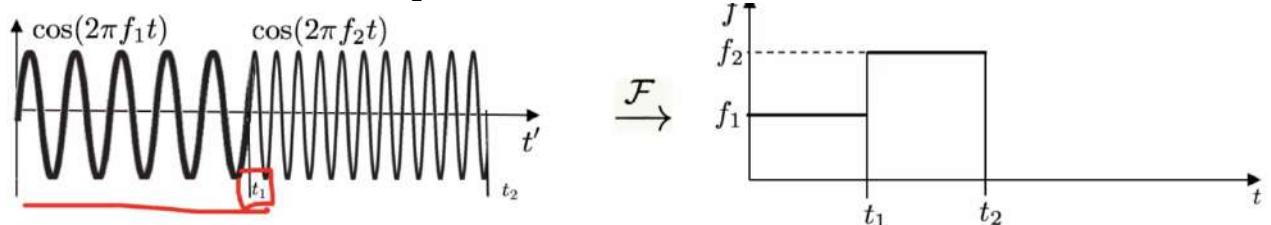
The phase of the sinusoids is the integral of the frequency that the sinusoid had at time  $t = T^{-1}(t')$ .

Let us consider the following example to better understand:

We have a signal that changes its frequency at a certain time  $2t_1$



If we apply the transformation  $t' = \frac{t}{2}$



We notice that from 0 up to  $t' = t_1$  I have the same sinusoid that in the original signal: the one with frequency  $f_1$  and amplitude 1. The same after  $t_1$ .

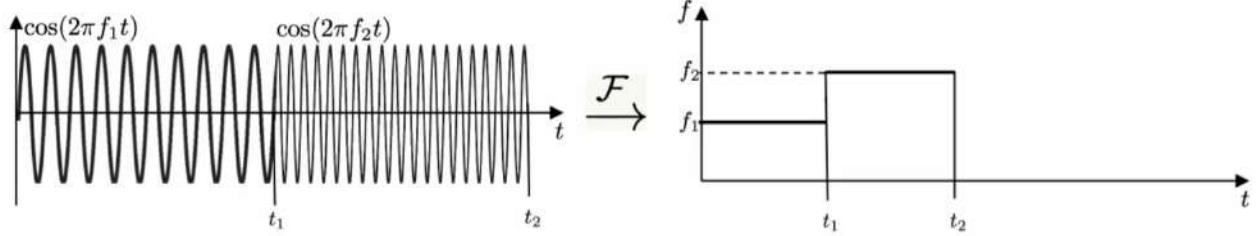
At the end, we changed the length of the sinusoid but not their frequency.

## Pitch Scaling

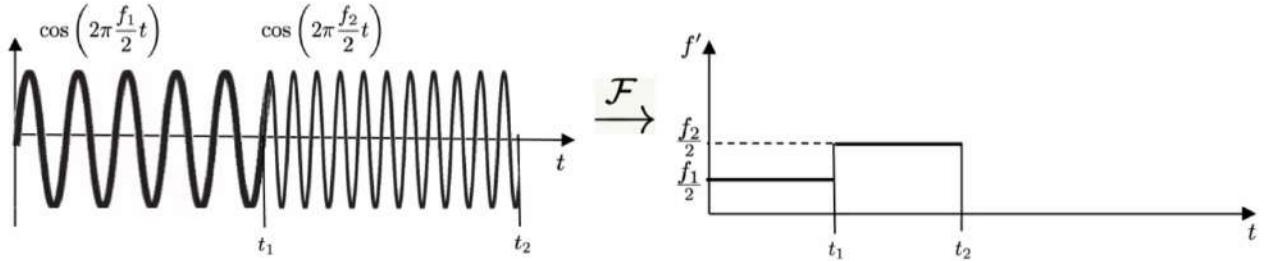
$$x'(t') = \sum_{i=1}^{I(T^{-1}(t'))} A_i(T^{-1}(t')) e^{j\phi'_i(t')} \quad \phi'_i(t') = \int_{-\infty}^{t'} \alpha(\tau) \omega_i(\tau) d\tau$$

In this case, I must not touch the temporal evolution. So what happens to  $t$  needs to be the same that happens to  $t'$ . What it changes is just the frequency: they are scaled of a factor  $\alpha(\tau)$ .

Thus, at instant  $t'$  I have a same number  $I(t')$  of sinusoids, but with different frequencies! Since the phase changes, also the frequencies change



We change the pitch by imposing  $f' = \alpha f = \frac{f}{2}$



What we do here is not drawing a sinusoid with different frequency. We draw point by point one sinusoid not asking myself which is the frequency of the sinusoid but by asking myself which is the phase of the sinusoid point by point. It is computed according to the frequency scaled so the frequency of the final sinusoid is also scaled of the same factor.

### Pitch and Frequency Scaling

Of course, we can also Mix and Match applying both temporal and frequency scaling.

$$x'(t') = \sum_{i=1}^{I(T^{-1}(t'))} A_i(T^{-1}(t')) e^{j\phi'_i(t')} \quad \phi'_i(t') = \int_{-\infty}^{t'} \alpha(T^{-1}(\tau)) \omega_i(T^{-1}(\tau)) d\tau$$

### Duality principle

An interesting thing is that you can apply time scaling by applying a combination of pitch scaling and resampling. Or you can apply pitch scaling by applying a combination of time scaling and resampling. Let us demonstrate it:

We start with ideal timescale modification

$$x'(t') = \sum_{i=1}^{I(T^{-1}(t'))} A_i(T^{-1}(t')) e^{j \int_{-\infty}^{t'} \omega_i(T^{-1}(\tau)) d\tau}$$

We now warp its time axis back substituting  $t'$  with  $T(t')$

$$y(t') \triangleq x'(T(t')) = \sum_{i=1}^{I(t')} A_i(t') e^{j \int_{-\infty}^{T(t')} \omega_i(T^{-1}(\tau)) d\tau}$$

---

<b>Warp back</b>	Distorcere, deformarsi su se stesso

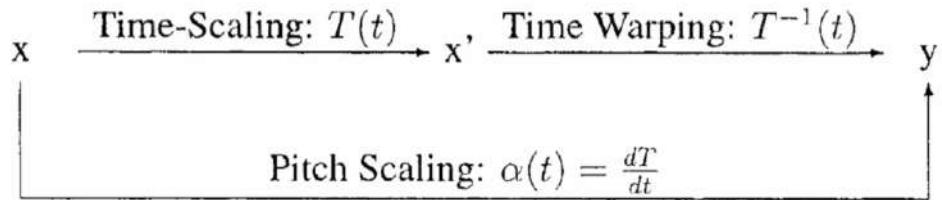
---

At the end, we perform a change of the integration variable  $\tau' = T^{-1}(\tau)$

$$y(t') \triangleq x'(T(t')) = \sum_{i=1}^{I(t')} A_i(t') e^{j \int_{-\infty}^{t'} (\frac{dT}{d\tau'}) \omega_i(\tau') d\tau'}$$

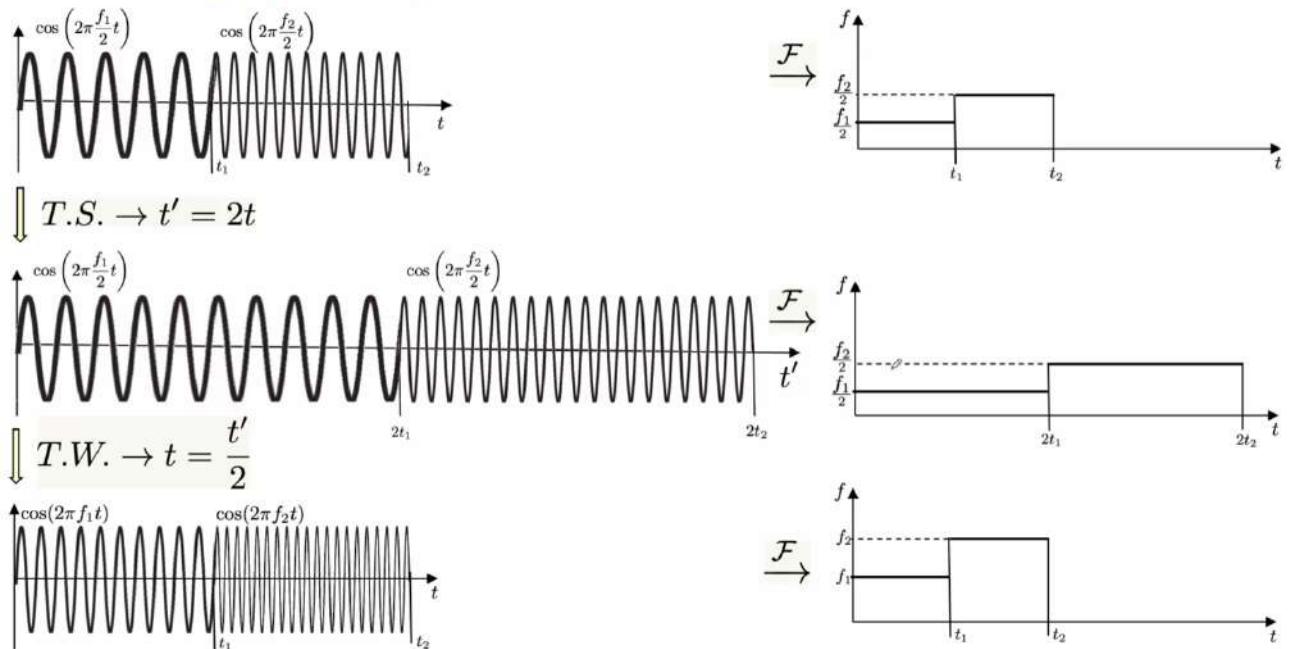
This corresponds exactly to a pitch scaling with  $\alpha(\tau) = \frac{dT}{d\tau'}$

We summarise the process in a graph:



What we conclude is, if we are able to perform either Pitch scaling or Time scaling, we are at the same time able to implement the other.

Graphically, what it happens is



The final signal has the same length of the original one, but the frequencies have changed. This is because in time scaling it does not change but in time warping it does. In fact, by changing the length of the time axis with time warping we change also the pitch.

### Methods to perform time and pitch scaling:

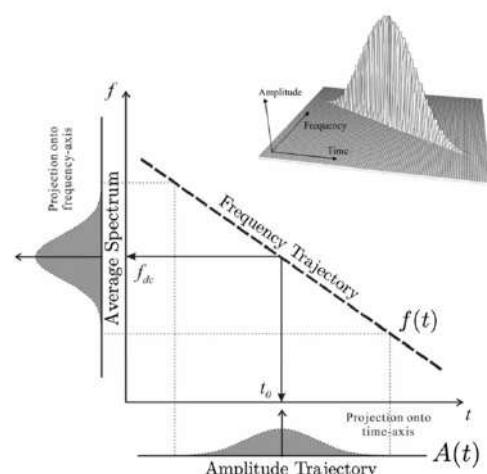
There are several methods to do that. The methods can be divided in two main classes: parametric methods and non-parametric methods.

#### *Parametric Frequency-based methods*

These are methods in which we can directly control the frequency parameters. If we think about it, they are nothing but what we have seen with synthesis by analysing by driving a bank of oscillators. Exactly the same.

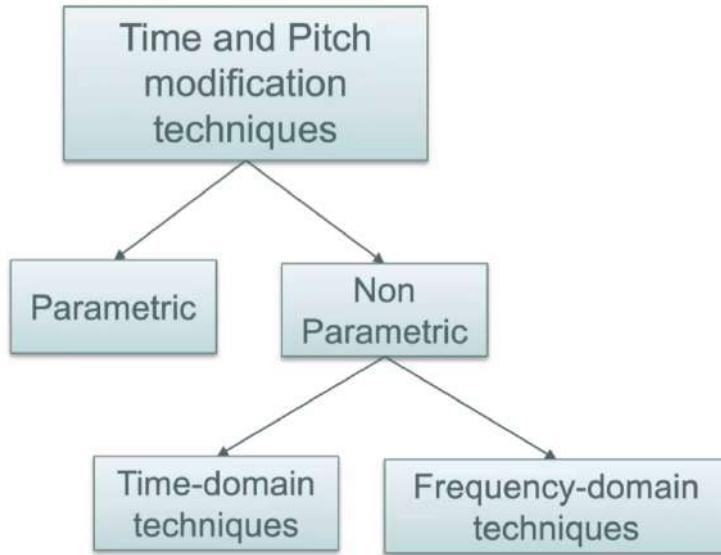
We summarise the process with an algorithm:

1. Analysis with STFT or filterbank;
2. Peak detection and tracking;
3. Modification of partials;
4. Synthesis with a bank of oscillators.



### Non-Parametric Frequency-based methods

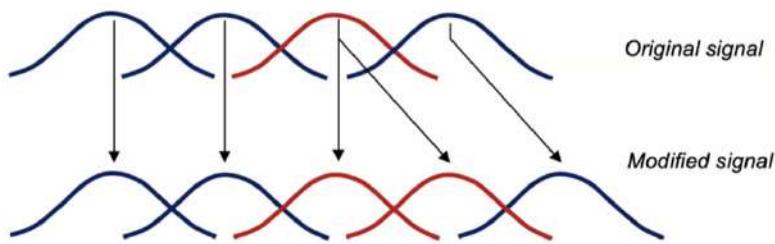
We will focus on non-parametric methos,



#### Time-domain non-parametric techniques (pitch-independent)

We start from time-domain parametric techniques. The idea is that we can breakdown the signal on which we want to apply time or pitch scaling and put everything together by discarding some of the windows or duplicate some of the windows.

So let us think about time scaling: we supposed to have an audio signal and we want to break it down into four frames using some overlap. Something that we can do is copying twice some of the parts of the signal.



By doing this we do not change the frequency of the sinusoid since we are not resampling discarding one sample every two! We are just changing the temporal evolution. If we use short segments about 20-40 ms, this method works! Furthermore, computationally speaking is extremely

cheap: we are applying a window, copy and paste. Nothing more.

## Analogue Pitch Scaling

The analogue pitch changing devices pre-dated their digital counterparts by several decades.

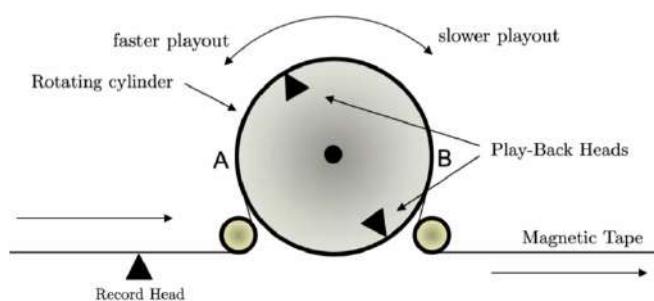
In *Duration and Frequency Alteration*, William Marlens traces the history of analogical pitch and time changing devices, with the earliest patents dating back to the 1920's. The basic idea of all of the patents was to record the audio signal onto some moving medium (tape, wire, film, etc.), and then use a rotating playback head, where the read heads would be moving at a different rate as the recording head.

The rotary playback head has 2, 4 or more read heads. By adjusting the rate of the rotation relative to the tape motion, the pitch of the signal can be raised or lowered.

Time expansion/compression can be achieved by speeding up or slowing down the rate of the tape, while keeping the tape heads moving past the tape at a rate that is identical to the original recording rate. As a given tape head comes into contact and is rotated away from the tape, the output signal from that given head will fade in and out, which results in a natural crossfading of pitch shifted segments.

The ELTRO was based on this idea and was mostly used to shorten audio for commercials to fit a given length, and for some occasional music and film productions. Some examples are following:

- Stockhausen: "Hymnen" <https://youtu.be/zDxpaxPMT0>
- Beach Boys: "She's Going Bald", starting at 0:51 you can hear the characteristic formant shift and warbly sound of a cross-fading pitch shifter.
- Stanley Kubrik's "2001 a space odissey" (voice of HAL 9000). The ELTRO was used on HAL's voice throughout the film, but the effect is most obvious in the computer deactivation scene <https://youtu.be/c8N72t7aScY> at 3:06, you can hear the voice start to warble more, presumably as the signal's pitch was shifted further downwards.



- If the cylinder rotates clockwise the head "follows" the tape, hence its relative speed is decreased. We are basically slowing down the reading experience.

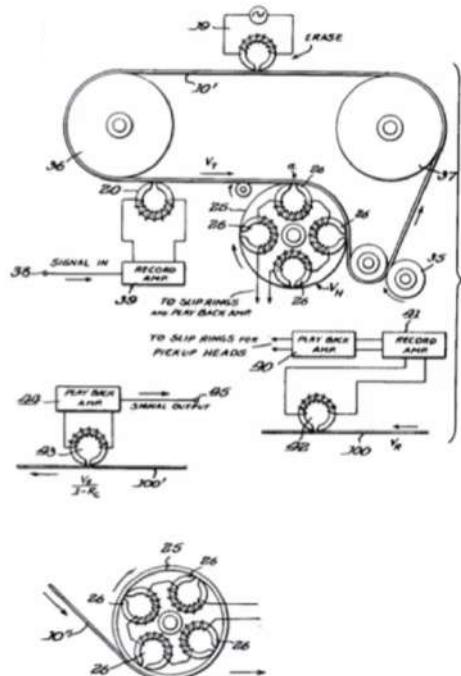
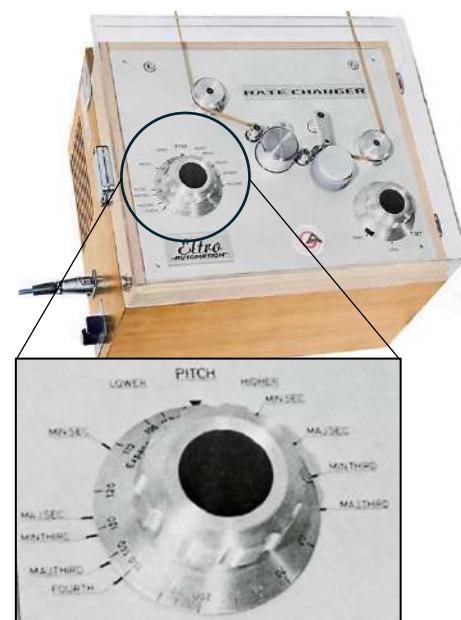


FIG. 10. Device of Fairbanks, Everitt and Jaeger, 1952.



Let us see more in detail how it works.

The key point is that we are reading at a different speed with respect to the one at which we are writing. The cylinder is rotating:

- If the cylinder rotates counter clockwise, against the direction of the tape, the relative speed between the rotation head and the tape is increased. We read faster than we are writing.

But speeding up or slowing down what are we doing? Time warping: if I read a tape at a faster speed it lasts less and the pitch goes up; the opposite happens if I read with a slower speed. There is a trick here! We consider the scenario in which the cylinder is rotating counter clockwise (speeding up). We have two heads, not just one.

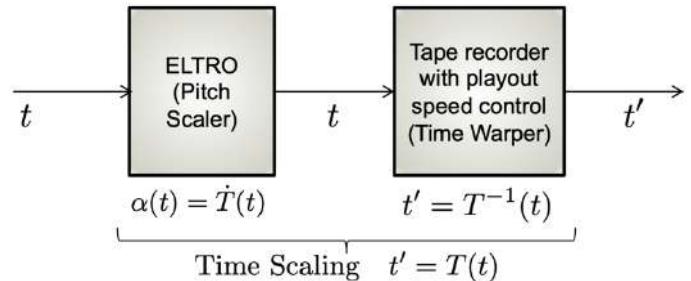
When the head is in correspondence of the point A, it loses the contact with the tape, while the head that is in correspondence of the point B touch the tape. In this way, the head in position B re-read the same portion that A has already read. Given the fact that we read two times each portion of tape, the duration remains the same! We are reading faster a longer tape. In this way we change just the duration but not the pitch. The opposite happens when the cylinder rotates clockwise.

### Analogical Time Scaling

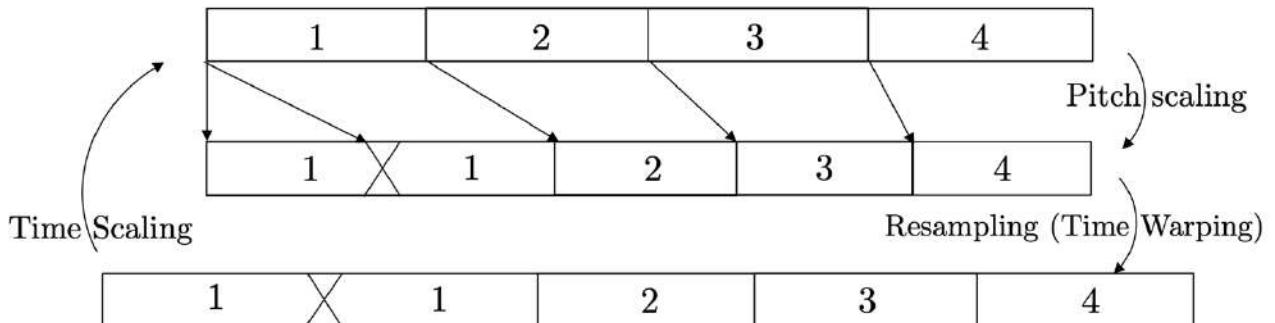
We have seen that when the cylinder rotates opposite to tape motion (counter clockwise), the playback is faster than recording, and the pitch is raised. When rotating clockwise, the playback is slower, and the pitch is lowered.

Assume the head is rotating counter clockwise: when a playback head leaves the

tape at point A, the other one comes in contact with the tape at point B, therefore the signal recorded on the tape between points A and B is repeated. In order to achieve time-scaling we need to resort to the duality theorem, i.e. use a second tape recorder, with different playout speed (time warping).



If we think in terms of windowed signal, we can see the thing in this way: the duration of the repeated segment is constant, and the splicing points are regularly spaced. As the heads gradually leave (or come into contact with) the tape at points A and B, their output signals gradually fade in or out (cross fading). The pitch can be lowered by making the cylinder rotate in the same direction as the tape motion. When the first play back head leaves the tape (point B), the signal recorded on the tape between points A and B will not be read by any playback head (discarded).



In Pitch scaling each portion of the signal lasts less and some of them are repeated to make the signal as long as the original. In time scaling, by resampling the pitch scaled signal we extend the duration of the segments and we make them longer but with the original pitch.

### Digital Counterpart

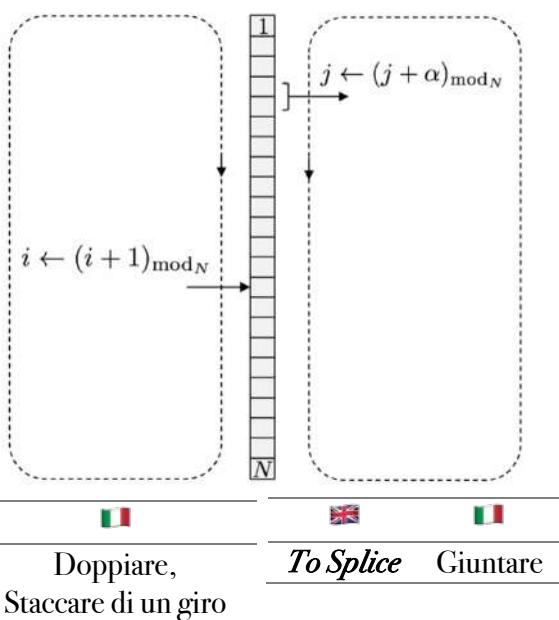
We can implement pitch scaling digitally by using a register and couple of pointers.

We need a register of pre-defined length N (circular buffer). The writing index  $i$  (record head) moves of integer steps (cycling through). The reading index  $j$  (record head) grows of fractional steps (cycling through), therefore an interpolation is needed.

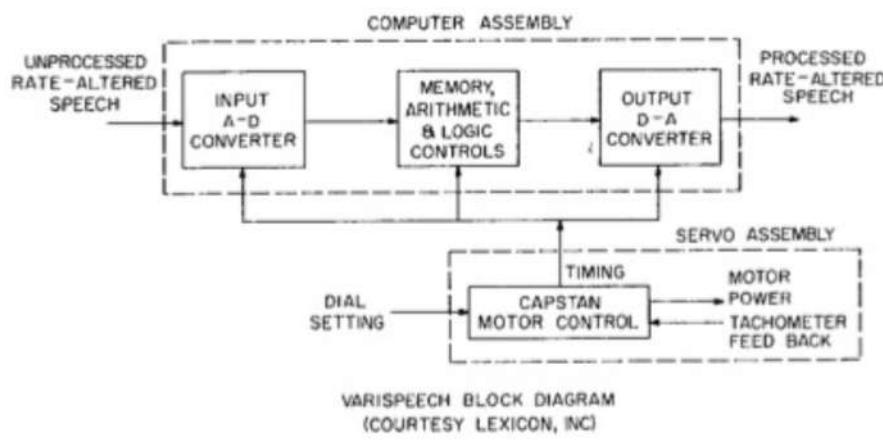
*Splicing*

- If  $\alpha > 1$ ,  $j$  runs faster than  $i$  (frequencies go up), therefore  $j$  periodically lap  $i$ . When this happens, we are reading the same splice (or part of it) twice.
- If  $\alpha < 1$ ,  $j$  runs slower than  $i$  (frequencies go down), therefore  $i$  periodically lap  $j$ . When this happens, we are writing new data before they can be read by the «playout head», which means that we are skipping a splice or part of it. We lose, therefore, part of the buffer

When an index passes the other, the output interpolation needs to take that into account and implemented some sort of cross-fading between the end of a splice and the beginning of the next one. To have a smooth transition from one sample to the other, we interpolate the previous with the following.



One remarkable circuit that implements this algorithm is the Lexicon Varispeech, produced in 1972.



We can hear at this link an example of how it sounds <https://youtu.be/eHgBZzdpTqo>

To see the firsts commercially available digital pitch shifter we have to wait for the Eventide H910 Harmonizer. I think that this is the Harmoniser that Giuni Russo wanted in the song “Un’estate al mare” when she sang: “Per regalo voglio un Harmoniser, con quel trucco che mi sdoppia la voce. Quest’estate ce ne andremo al mare per le vaaaaacCAAAANZEEEEEE! UUUUN ESTATE AL MAREEEEEEH!”

We can watch its story narrated by Tony Visconti and others at this link

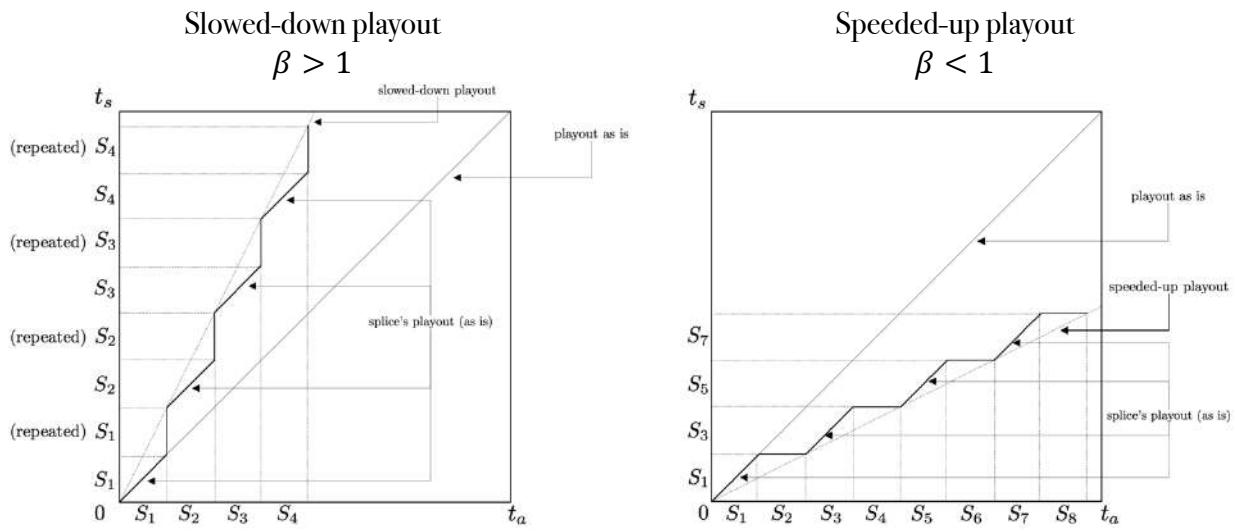
<https://www.youtube.com/watch?v=977Sri5EcCE>.

We can hear its effect in the intro of Luka by Suzanne Vega, in the intro of Back in black by ACDC and in many other songs.



### Scaling in time domain: I/O time mapping

As we saw, we are basically replicating or deleting part of the audio track. It is as if we would use a weird input-output mapping function. On the x axis we have the time we are analysing ( $t_a$ ) while on the y axis we have the synthesis time ( $t_s$ ) so the signal that we are reading  $t_s = \beta t_a$ . The dashed line represents the ideal time scaling, playout as it is. The piecewise linear curve represents the playout based on splicing.



We notice how the playout periodically repeats input splices, or parts thereof

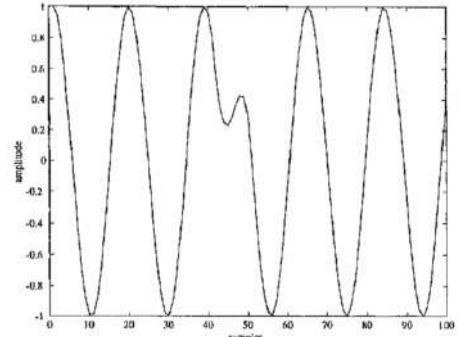
On the other hand, in this case the playout periodically skips input splices, or part of thereof

### Issues:

We can imagine that the fundamental issue is how to connect two frames. It is almost impossible not to create inconsistencies that, in turns, create artifacts. In fact, cross fading guarantees continuity, but does not eliminate artefacts due to misalignment.

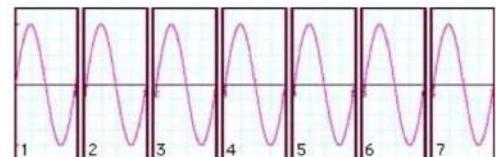
How can we improve it?

By using another method that exploits wave similarities.



### Time-domain non-parametric techniques (waveform similarity)

The idea is, if we know that the signal is periodic or nearly periodic and we break the signal down in multiple frames, if the frames take into account a multiple of the observed period of the signal, we probably make a better job than taking into account random portions of the signal.



With this in mind, we look for some methods that allow us to divide the signal to perform time scaling in a smarter way.

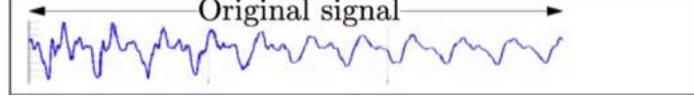
There are two main methods:

1. SOLA – Synchronous Overlap and Add
2. PSOLA – Pitch-Synchronous Overlap and Add

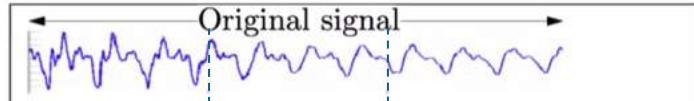
### SOLA:

The idea is why should I fix the starting point and the ending point of all my windows to be always the same? Why cannot I adjust the position of the analysis frames? Digitally, we can! We put this to practice. We suppose to want apply time scaling by replicating some of the windows.

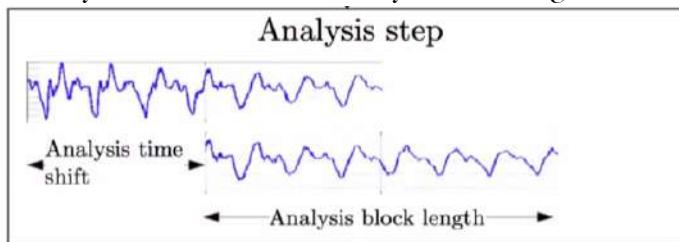
We have a certain signal:



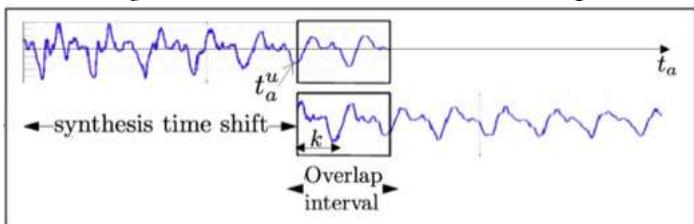
We divide it in blocks:



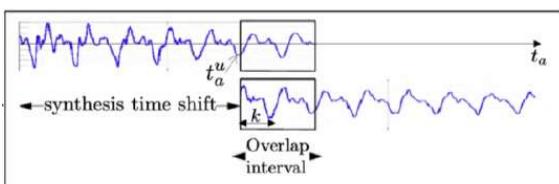
We analyse it by finding the analysis time shift and the analysis block length:



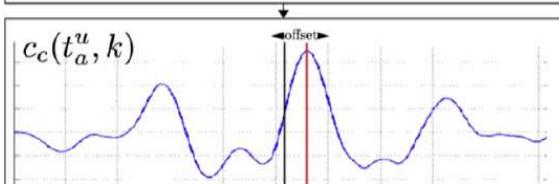
At this point we are ready to perform the synthesis. We copy the same window considering a certain shift in time in order to perform time scaling, and then we cross-fade the two samples that we have to overlap.



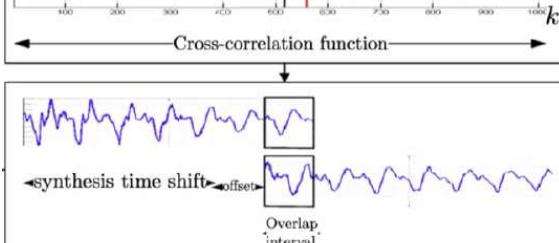
At this point we ask ourselves: the overlapped parts match well?



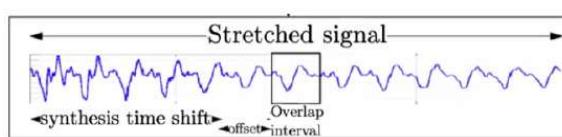
If no, we take another window by slightly moving the time instant from which the window starts and, then, we check again if it overlaps well or not.



What we are doing is somehow synchronising this overlap and add so that the two samples match properly.



What we obtain at the end is the synthesised signal with a certain time offset generated by this adjustment.



But, what does “match well” mean? We have to define one way of measuring if the overlap is good. Formalising our aim, we need to find a measure that express how much the overlapped parts are maximally similar. Cross-correlation could be a good solution. Since we want to make the correlation signal independent, we normalise it with respect to the input signal.

$$c_c(t_a^u, k) = \frac{\sum_{i=1}^{N_c} x(t_a^u + i)x(t_a^u + i + k)}{\sqrt{\sum_{i=1}^{N_c} x^2(t_a^u + i)} \sqrt{\sum_{i=1}^{N_c} x^2(t_a^u + i + k)}}$$

We check the correlation of the singal with itslef when we skipped  $k$  samples. We try different  $k$  until we find the value of  $k$  that maximise the correlation functions. Mathematically speaking

$$k_s = \operatorname{argmax}_k [c_c(t_a^u, k)]$$

As an alternative, we can find the offset  $k$  for witch the two samples match well as the value that minimise the average magnitude difference function (AMDF); function that can be expressed by

$$f_c(t_a^u, k) = \frac{1}{N_c} \sum_{i=1}^{N_c} |x(t_a^u + 1) - x(t_a^u + i + k)|$$

Where we look for:

$$k'_s = \operatorname{argmin}_k [f_c(t_a^u, k)]$$

From what we have just said, we could affirm that the offset  $k$  by whitch to obtain the perfect match depends on the measuirng function that we take into account. This is theoretically true, but  $k_s$  and  $k'_s$  are very similar one to eachother in practice.

In fact

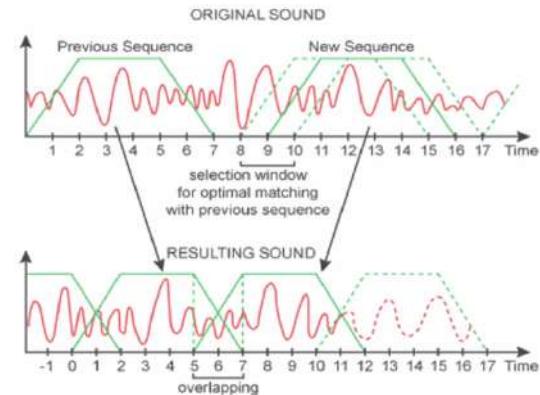
$$\begin{aligned} k'_s &= \operatorname{argmin}_k \{f_c(t_a^u, k)\} \\ &\stackrel{\substack{\equiv \\ f_c(t_a^u, k) = \frac{1}{N_c} \sum_{i=1}^{N_c} |x(t_a^u + 1) - x(t_a^u + i + k)|}}{=} \\ &= \operatorname{argmin}_k \left\{ \frac{1}{N_c} \sum_{i=1}^{N_c} |x(t_a^u + 1) - x(t_a^u + i + k)| \right\} \quad \text{For small values of } x \\ &\quad |x| \approx x^2 \end{aligned}$$

	$f : y =  x $
	$g : y = x^2$

$$\cong \operatorname{argmin}_k \left\{ \frac{1}{N_c} \sum_{i=1}^{N_c} (x(t_a^u + 1) - x(t_a^u + i + k))^2 \right\} =$$

$$\begin{aligned}
&= \operatorname{argmin}_k \left\{ \frac{1}{N_c} \sum_{i=1}^{N_c} [x^2(t_a^u + 1) + x^2(t_a^u + i + k) - 2x(t_a^u + 1)x(t_a^u + i + k)] \right\} = \\
&= \operatorname{argmin}_k \left\{ \frac{1}{N_c} \sum_{i=1}^{N_c} x^2(t_a^u + 1) + \frac{1}{N_c} \sum_{i=1}^{N_c} x^2(t_a^u + i + k) - \frac{2}{N_c} \sum_{i=1}^{N_c} x(t_a^u + 1)x(t_a^u + i + k) \right\} \underset{k \text{ is small}}{\underset{\approx}{=}} \\
&\approx \operatorname{argmin}_k \left\{ \frac{1}{N_c} \sum_{i=1}^{N_c} x^2(t_a^u + 1) + \frac{1}{N_c} \sum_{i=1}^{N_c} x^2(t_a^u + i) - \frac{2}{N_c} \sum_{i=1}^{N_c} x(t_a^u + 1)x(t_a^u + i + k) \right\} \underset{E_x = \frac{1}{N_c} \sum_{i=1}^{N_c} x^2(t_a^u + 1)}{\underset{\approx}{=}} \\
&= \operatorname{argmin}_k \left\{ E_x + E_x - \frac{2}{N_c} \sum_{i=1}^{N_c} x(t_a^u + 1)x(t_a^u + i + k) \right\} = \\
&= \operatorname{argmin}_k \left\{ 2E_x - \frac{2}{N_c} \sum_{i=1}^{N_c} x(t_a^u + 1)x(t_a^u + i + k) \right\} \underset{\substack{\text{We throw away all that is} \\ \text{not useful to find the point} \\ \text{of minimum:}}}{{\underset{\approx}{=}}} \\
&\quad \underset{-E_x \text{ does not depend on } k}{\underset{-2 \text{ and } \frac{1}{N_c} \text{ are just constant}}} \\
&= \operatorname{argmin}_k \left\{ - \sum_{i=1}^{N_c} x(t_a^u + 1)x(t_a^u + i + k) \right\} \underset{\substack{\text{The point of minimum} \\ \text{of a function} \\ \text{is equal to the point of} \\ \text{maximum of that function} \\ \text{flipped}}}{{\underset{\approx}{=}}} \\
&\quad \underset{\operatorname{argmin}_k \{x\} = \operatorname{argmax}_k \{-x\}}{} \\
&= \operatorname{argmax}_k \left\{ \sum_{i=1}^{N_c} x(t_a^u + 1)x(t_a^u + i + k) \frac{\sqrt{\sum_{i=1}^{N_c} x^2(t_a^u + i)}}{\sqrt{\sum_{i=1}^{N_c} x^2(t_a^u + i)}} \frac{\sqrt{\sum_{i=1}^{N_c} x^2(t_a^u + i + k)}}{\sqrt{\sum_{i=1}^{N_c} x^2(t_a^u + i + k)}} \right\} = \\
&= \operatorname{argmax}_k \left\{ c_c(t_a^u, k) \sqrt{\sum_{i=1}^{N_c} x^2(t_a^u + i)} \sqrt{\sum_{i=1}^{N_c} x^2(t_a^u + i + k)} \right\}
\end{aligned}$$

At the end, what we are doing is shifting the window that selects the new sequence that we have to overlap-and-add to the previous one by a number of samples  $k_s$  that minimise the normalised cross correlation (or  $k'_s$  that normalise the AMDF) in order to find the best match. SOLA is basically the Tinder of frames: it uses maths to find the supposed (based on statistical parameters) perfect match.

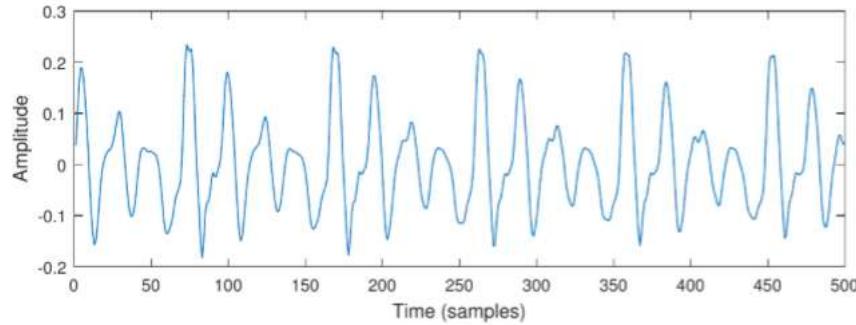


### PSOLA:

Pitch synchronous overlap and add looks a lot like Synchronous overlap and add, so we do not touch anything inside the window. But surprise surprise, we perform pitch scaling!

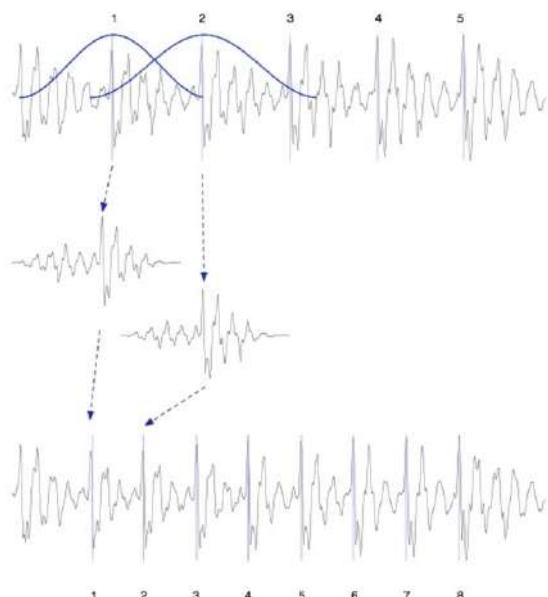
Reminder: Speech signal.

A speech signal can be considered as a periodic signal processed by a special filter that applies formants. In time domain it is represented as follows:



As we can notice the amplitude is modulated creating a periodical pattern. Simply speaking:

- The pattern gives the vowel;
- The periodicity of the pattern gives the pitch.



PSOLA is based on the assumption that the local value of the pitch is known as well as the locations of glottal pulses (peaks of pressure resulting from the rapid closure of the vocal folds). Pitch-scale modifications consists of extracting short-time segments of signal centred around the successive glottal pulses and adding them together at a different rate by re-windowing. This does not shift the formants, only the density of glottal pulses since we are basically changing the distance (so the period) between them.

If we make them closer, we reduce the period, so we increase the frequency; if we make them larger, we increase the period and we decrease the frequency.

The most important here, to obtain the correct pitch is that the spacing between the replica, in terms of synthesis,

remains constant at the pitch that we want. We can modify a bit the original frame to obtain a smoother cross-fade but, what it is most important, is the period between them.

Of course, it can be used also for time scaling because of the duality principle. Rather than coping frames as they are, we stretch them by re-windowing them.

This type of system works properly just for small changes of pitch. If we change the pitch too much it does not work since we modify too much the pattern with overlapping. Also, the duality principle does not work really well: if we apply straightforward the duality principle, with the warping we distort the formants. So it is better using simple overlap and add for time scaling to play with replicating or deleting some of the windows rather than applying the duality principle. As a rule of thumb, we say that PSOLA works when we change the pitch from 0.5 to 2 times.

In principle we can map the time samples in a virtual time domain using complex time transformation.

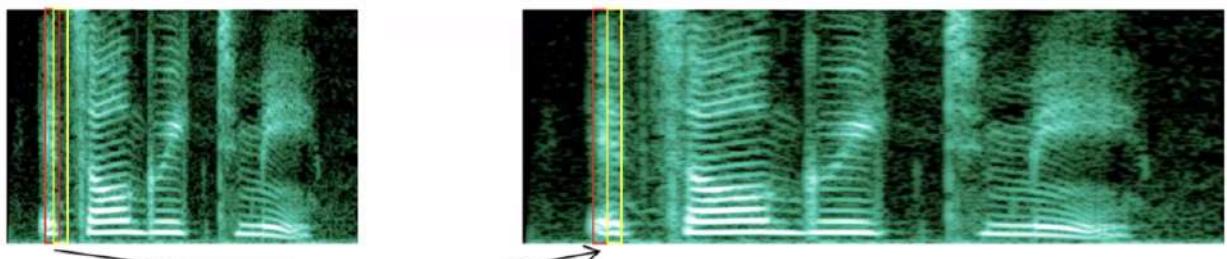
Anyway, the simplest thing that we can do is:

1. Set up the synthesis train of pulses (source).
2. Go back to the analysis signal and find the periodicity of it.
3. Checking, on the synthesis train of pulses, which is the closest input pulse that I have.
4. Copying it.

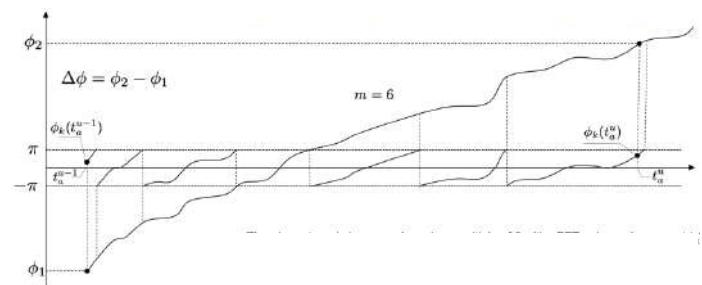
Some final remarks on these methods: they are simple, they are fast, so they are computationally inexpensive. However, they perform well just in particular situation in which the modification factors are very small. Time or pitch scale modifications by large factors are not suitably implemented with time domain methods and usually require more elaborated frequency-domain techniques. Furthermore, they do not work well in the presence of heavy reverberations, echoes or background noise. In this case it is impossible not to have artefacts. In order to avoid them we came up with the next method.

#### Frequency-domain non-parametric techniques (STFT-based)

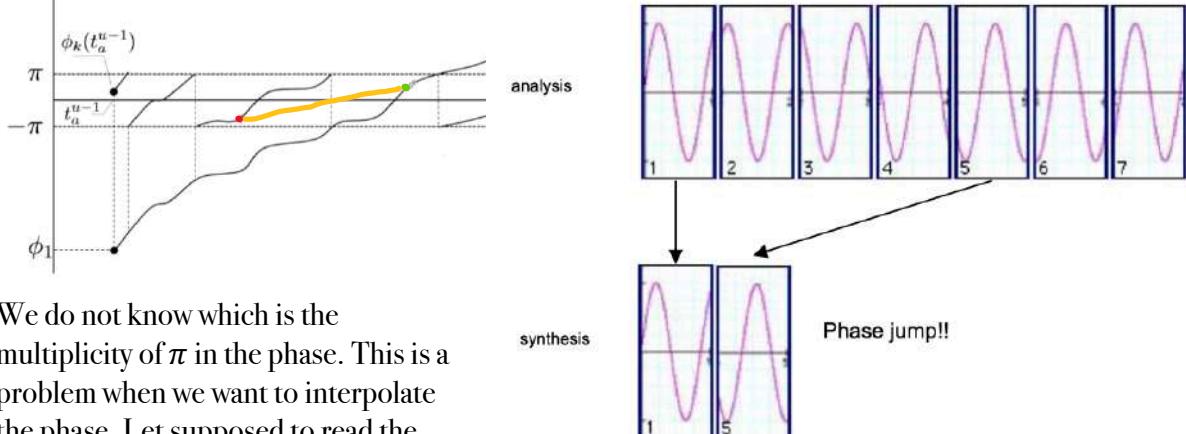
This kind of method is a bit more complex even though the idea is pretty simple: we can interpret the spectrogram (amplitude of the STFT) as an image; to apply time scaling we just have to stretch this image horizontally.



The frequency behaviour we have in the picture on the right is the same we have in the picture on the left. This is true. But, what about the phase? We know that to come back to time domain we need surely the amplitude but also the phase: we need to find a way to build it. In order to make it, we have to interpolate the phase, operation that is extremely tricky. In fact, when we compute the DTFT or the STFT, we wrap the phase. This means that all the values of the phase will be included into the interval  $[-\pi; \pi]$  independently to the behaviour of the phase.



We get this result by definition. We remind that the phase is defined as the  $\text{atan2}\left(\frac{\text{Im}}{\text{Re}}\right)$  and that the codomain of this function varies from  $-\pi$  to  $\pi$ .

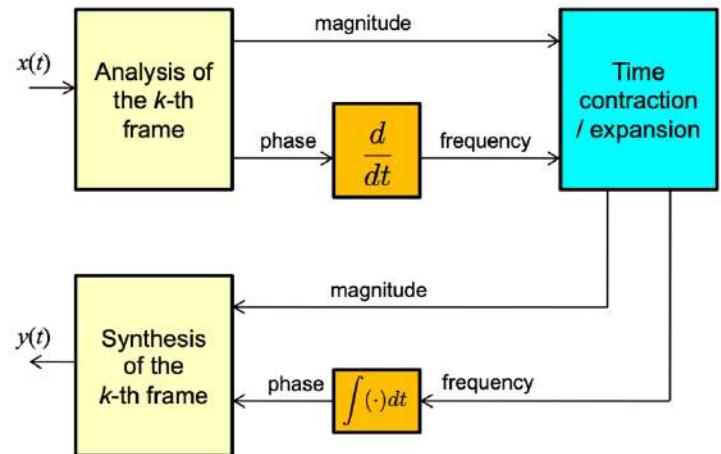


We do not know which is the multiplicity of  $\pi$  in the phase. This is a problem when we want to interpolate the phase. Let supposed to read the value marked by the red point for a certain frame and the value marked by the green one for another. We try now to interpolate them. What we obtain is the yellow line that does not correspond to the actual phase. When we read the phase of a sinusoid on that yellow line, we find a value that does not correspond to the actual one. Another way to see the problem is that by connecting the phase points through the yellow line, I completely misjudging the frequency of the signal since the slope of the phase changing in time is the frequency of the signal. We must understand how to unwrap the phase correctly. Once we understand how to find the actual line in phase when we interpolate, the problem is solved.

All this translates into a phase jump in the synthesis signal.

So, the pipeline we have to follow is this one.

- 1) Analysis:  
Compute the STFT at constant rate (magnitude and phase)
- 2) Scaling
  - Derive **magnitude** and **frequency** of each spectral component
  - Warp time axis
  - Compute new phase information through integration of frequencies on warped time axis (need phase unwrapping)
- 3) Synthesis
  - Given the sequence of FFTs, reconstruct scaled signal through inverse STFT



So, we fill the gap finding a way to interpolate the phase in our window. We consider the usual signal model

$$x(t) = \sum_{i=1}^{I(t)} A_i(t) e^{j\phi_i(t)} \quad \text{with} \quad \phi_i(t) = \int_{-\infty}^t \omega_i(\tau) d\tau$$

In principle there is no guarantee that given any sequence of FFTs we can go back in time-domain. We assume it that we can do it for whatever signal.

So, given the analysis equation

$$X(t_a^u, \Omega_k) = \sum_n h(n)x(t_a^u + n)e^{-j\Omega_k n} \quad \Omega_k = \frac{2k\pi}{N}$$

We assume that we can always go back in time domain by using the following synthesis equation

$$x(n) = \sum_u w(n - t_a^u) \frac{1}{N} \sum_{k=0}^{N-1} X(t_a^u, \Omega_k) e^{j\Omega_k(n-t_a^u)}$$

Being fair, the STFT can be reversed if and only if  $\sum_u w(n - t_a^u)h(t_a^u - n) = 1$   
Anyway, we substitute our function into the analysis equation

$$X(t_a^u, \Omega_k) = \sum_n h(n) \left( \sum_{i=1}^{I(t_a^u+n)} A_i(t_a^u + n) e^{j\phi_i(t_a^u+n)} \right) e^{-j\Omega_k n}$$

We make the hypothesis of stationary condition within the window. It means that for a given time window, the phase changes linearly with respect to the frequency, the amplitude and the phase remains constant.  
Therefore

$$\begin{aligned} X(t_a^u, \Omega_k) &= \sum_n h(n) \left( \sum_{i=1}^{I(t_a^u+n)} A_i(t_a^u + n) e^{j\phi_i(t_a^u+n)} \right) e^{-j\Omega_k n} \\ &\stackrel{\substack{A_i(t_a^u+n)=A_i(t_a^u) \\ I(t_a^u+n)=I(t_a^u) \\ \phi_i(t_a^u+n)=\phi(t_a^u)+n\omega_i(t_a^u)}}{=} \\ &= \sum_n h(n) \left( \sum_{i=1}^{I(t_a^u)} A_i(t_a^u) e^{j[\phi(t_a^u)+n\omega_i(t_a^u)]} \right) e^{-j\Omega_k n} = \\ &= \sum_n h(n) \sum_{i=1}^{I(t_a^u)} A_i(t_a^u) e^{j\phi(t_a^u)} e^{jn\omega_i(t_a^u)} e^{-j\Omega_k n} = \\ &= \sum_n h(n) \sum_{i=1}^{I(t_a^u)} A_i(t_a^u) e^{j\phi(t_a^u)} e^{-jn[\Omega_k - \omega_i(t_a^u)]} = \\ &\stackrel{\substack{H(\omega)=\sum_n h(n)e^{-jn\omega}}}{=} \\ &= \sum_{i=1}^{I(t_a^u)} A_i(t_a^u) e^{j\phi(t_a^u)} H(\Omega_k - \omega_i(t_a^u)) \end{aligned}$$

We make a second hypothesis, the Resolution condition hypothesis.

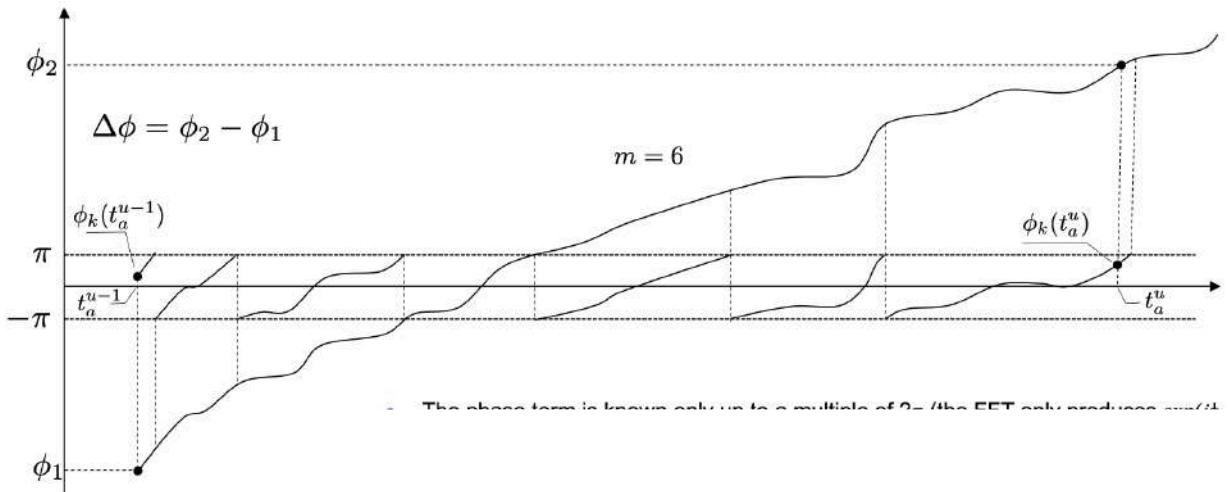
- We assume  $h(n)$  to be real and symmetric about  $n = 0$ , so that  $H$  is real and symmetric
- We also select the cut-off frequency  $\omega_h$  of  $h(n)$  to be less than half the spacing between two successive sinusoids, so that for each frequency bin just one sinusoid exists.

This means that we can re-write the equation with something easier

$$\begin{aligned} &= \sum_{i=1}^{I(t_a^u)} A_i(t_a^u) e^{j\phi_i(t_a^u)} H(\Omega_k - \omega_i(t_a^u)) = \\ &= \begin{cases} A_k(t_a^u) e^{j\phi_k(t_a^u)} H(\Omega_k - \omega_k(t_a^u)), & \text{if } |\Omega_k - \omega_k(t_a^u)| \leq \omega_h \\ 0, & \text{otherwise} \end{cases} \end{aligned}$$

This tells us that the STFT gives access to the instantaneous amplitude  $A_k$ , and to the instantaneous phase  $\phi_k$  of the  $k$ -th sinusoid that falls into Fourier channel  $k$ . From this equation here I can read point by point for each frequency-time pair the amplitude of the sinusoid there and the phase of that sinusoid.

Since the phase is wrapped between  $-\pi$  and  $\pi$ , if I measure the difference of phase I find the wrong value we were talking about before so  $\Delta\phi_w = \phi_k(t_a^u) - \phi_k(t_a^{u-1})$ .



The actual phase, the one we want.

How can we outfox the problem?

By simply estimating it taking into account the number of phase wrappings  $m$ :

$$\Delta\phi = \phi_2 - \phi_1 = \Delta\phi_w + m \cdot 2\pi \quad \stackrel{\text{assuming the frequency}}{=} \quad (t_a^u - t_a^{u-1})\omega_k(t_a^u) \quad \stackrel{R=(t_a^u - t_a^{u-1})}{=} \quad R\omega_k(t_a^u)$$

From which we obtain

$$\omega_k(t_a^u) = \frac{\Delta\phi_w + m \cdot 2\pi}{R}$$

But how can we find  $m$ ?

As the sinusoid falls within the  $k$ -th channel, thanks to the resolution condition (hypothesis 2) we have

$$|(\omega_k(t_a^u) - \Omega_k)R| \leq \omega_h R$$

Where  $\omega_h$  is the cut-off frequency of the window defined in hypothesis 2.

Time to make one last hypothesis: we choose the parameter in a way that  $\omega_h R < \pi$ . Therefor

$$\begin{aligned} |(\omega_k(t_a^u) - \Omega_k)R| \leq \omega_h R < \pi \rightarrow |(\omega_k(t_a^u) - \Omega_k)R| < \pi \\ \xrightarrow{\omega_k(t_a^u) = \frac{\Delta\phi_w + m \cdot 2\pi}{R}} \\ \rightarrow \left| \left( \frac{\Delta\phi_w + m \cdot 2\pi}{R} - \Omega_k \right) R \right| < \pi \rightarrow |\Delta\phi_w + m \cdot 2\pi - \Omega_k R| < \pi \end{aligned}$$

And there is only one integer  $m$  that satisfies this last inequality!

How can we guarantee this condition?

The cut-off frequency of the window is half its main lobe, so we can write the following equation

$$2\omega_h = L \frac{2\pi}{NT}$$

With

- $L$  window factor
- $N$  length of the window
- $T$  sampling period

By simplifying the factor 2 and multiplying by  $R$  on both sides, we have the product  $\omega_h R$  on the left that we can limit to  $\pi$ .

$$\omega_h = L \frac{\pi}{NT} \rightarrow \omega_h R = L \frac{\pi}{NT} R \rightarrow \omega_h R = L \frac{\pi}{NT} R < \pi$$

At the end we obtain that

$$L \frac{\pi}{NT} R < \pi \rightarrow \frac{R}{T} < \frac{N}{L}$$

Where we have, on the left,  $\frac{R}{T}$  that expresses how many samples there are between the starting point of one window and the starting point of the next (the hop size); on the right  $\frac{N}{L}$  so the samples of the window divided by the factor that characterised the type of window. It express the percentage of overlap that needs to be from one sample to the other. Since the smoother the window, the higher the  $L$ , the smoother the window the higher will be the overlap.

Once we have  $m$  we can find the instantaneous frequency as

$$\omega_k(t_a^u) = \frac{\Delta\phi_w + m \cdot 2\pi}{R}$$

Let us summarise everything with an algorithm:

- 1) Set the initial instantaneous phases

$$\hat{\phi}(0, \Omega_k) = \angle X(0, \Omega_k)$$

- 2) Set the next synthesis and analysis instant to

$$t_s^{u+1} = t_s^u + R \quad \text{and} \quad t_a^{u+1} = T^{-1}(t_s^{u+1})$$

- 3) Compute the instantaneous frequency in each channel as time  $t_a^{u+1}$  as

$$\omega_k(t_a^u) = \Omega_k + \frac{1}{R(u)} (\Delta\phi - \Omega_k R(u) - 2m\pi)$$

- 4) Compute the instantaneous phase using

$$\hat{\phi}_k(t_s^u) = \hat{\phi}_k(t_s^{u-1}) + (t_s^u - t_s^{u-1})\omega_k(T^{-1}(t_s^u))$$

- 5) Reconstruct the time-scaled STFT using

$$Y(t_s^u, \Omega_k) = |X(T^{-1}(t_s^u), \Omega_k)| e^{j\hat{\phi}_k(t_s^u)}$$

- 6) Compute the  $(u + 1)$  term of the output (inverting the STFT)

$$y(n) = \sum_u w(n - t_s^u) \frac{1}{N} \sum_{k=0}^{N-1} X(t_s^u, \Omega_k) e^{j\Omega_k(n - t_s^u)}$$

- 7) Set  $u \leftarrow u + 1$

- 8) Go back to step 2

Just by looking at this algorithm, we can imagine that the computational burden is not that low.

And what about **Pitch Scaling in frequency domain?**

In order to implement it, we can use the duality theorem (by combining time-scaling and resampling, so time warping).

## References

To deepen some of these aspects, we can read the Chapter 7 of the book “Applications of digital signal processing to audio and acoustics” by Kahrs, Brandenburg.

## Matlab's lesson upon pitch and time rescaling.

### Exercise 1

Write a simplified version of PSOLA algorithm:

1. Load the A String.wav file and play it
2. Detect its fundamental period  $T_a$  through Fourier analysis
3. Using Hann window and a pitch-scaling factor  $\alpha = 0.9$  synthesize a pitch scaled version of the track by moving windows of signals spaced by  $T_a$  of  $T_a * \alpha$
4. Play the obtained signal

```
%%%%%%
% PSOLA
% SASP / DAAP
% 2024
% %%%%%%
clc, clear all, close all

%% Read audio track
% Read signal
[y, fs] = audioread('A string.wav');
N = length(y); % Signal length in samples
Ts = 1/fs; % Sampling period

% Convert to mono if stereo
y = y(:, 1);

% Plot waveform
t = 0:1/fs:(N-1)/fs;
figure(1)
plot(t, y)
xlabel('time [s]')

%% Detect fundamental frequency from Fourier analysis
% Compute FFT magnitude
Y = abs(fft(y));

% Plot the spectrum
f = linspace(0, fs, N+1); % Define frequency axis
f = f(1:end-1);

figure(2), hold on
plot(f, Y);
xlabel('freq [Hz]')
grid('on')

% Detect fundamental frequency of the harmonic sound
% hint: check 'max' and 'findpeaks' functions
[Y_max, Y_loc] = findpeaks(Y, 'MinPeakHeight', 0.3 * max(abs(Y))); % Only peaks higher than 30% of the maximum value
Y_max = Y_max(1); % Select first peak
Y_loc = Y_loc(1); % Select first peak location
f_fund = Y_loc * fs / N; % Compute fundamental frequency
plot(f_fund, Y_max, '*'); % Plot peak position
```

This function automatically finds the peaks in a function.

```

%% Generate analysis train of pulses
% Compute the period associated to the fundamental frequency
T_a = 1/f_fund;
T_a_samples = round(T_a*fs);

% Generate a train of pulses according to the computed period
train_a = zeros(N, 1); % Start from a vector of zeros
train_a(1:T_a_samples:N-T_a_samples) = 1; % Write 1 every T_a_samples to the end

% Compute pulses locations
idx_a = find(train_a == 1); % Find position of elements equal to 1

%% Generate synthesis train of pulses
% Choose a pitch scaling factor
alpha = .9;

% Compute the new period associated to the scaling factor
T_b_samples = round(T_a_samples * alpha);

% Generate a train of pulses according to the computed period
train_b = zeros(N, 1); % Start from a vector of zeros
train_b(1:T_b_samples:N-T_b_samples) = 1; % Write 1 every T_b_samples to the end

% Compute pulses locations
idx_b = find(train_b == 1); % Find position of elements equal to 1

%% Plot both trains of pulses
figure(3)
subplot(2,1,1), stem(train_a(1:3000)), title('Analysis peaks')
subplot(2,1,2), stem(train_b(1:3000)), title('Synthesis peaks')

%% Window
% Generate a Hann window with the correct lenght for synthesis
L = T_b_samples * 2 + 1; % In order to honor overlap and add reconstruction condition
win = hann(L);

% Check windows ovrelapping condition
figure(4)
plot(conv(win, train_b), 'LineWidth', 3) % All overlapped windows add to 1

%% PSOLA (simplified)
% Allocate memory for synthesis signal
y_out = zeros(N + 2*L, 1); % Big enough buffer (can be optimized)

% Loop over all synthesis windows
for n = 1:length(idx_b)

    % Select synthesis window position
    start_out = idx_b(n) - floor(L/2); % Start synthesis sample index
    stop_out = idx_b(n) + floor(L/2); % End synthesis sample index

    % Select analysis window

```

We add +1 since we want an odd number.

```

[~, n_a] = min(abs(idx_a - idx_b(n))); % Find position of the closest
analysis peak given the chosen synthesis one

start_in = idx_a(n_a) - floor(L/2); % Start analysis sample index
stop_in = idx_a(n_a) + floor(L/2); % End analysis sample index

% Copy analysis window to synthesis position
if start_in >= 0 && start_out >= 0 && stop_in <= length(y) && stop_out <=
length(y_out) % If we are not writing outside of the buffer or reading
outside of the analysis signal
    x = y(start_in:stop_in) .* win; % Select the analysis signal window
    y_out(start_out:stop_out) = y_out(start_out:stop_out) + x; % Copy it
over the correct synthesis part of the buffer using overlapp and add
end

end

%% Play original and pitch-shifted signals
soundsc(y/max(y), fs) % Original track
pause(6)
soundsc(y_out/max(y_out), fs) % Pitch shifted track

```

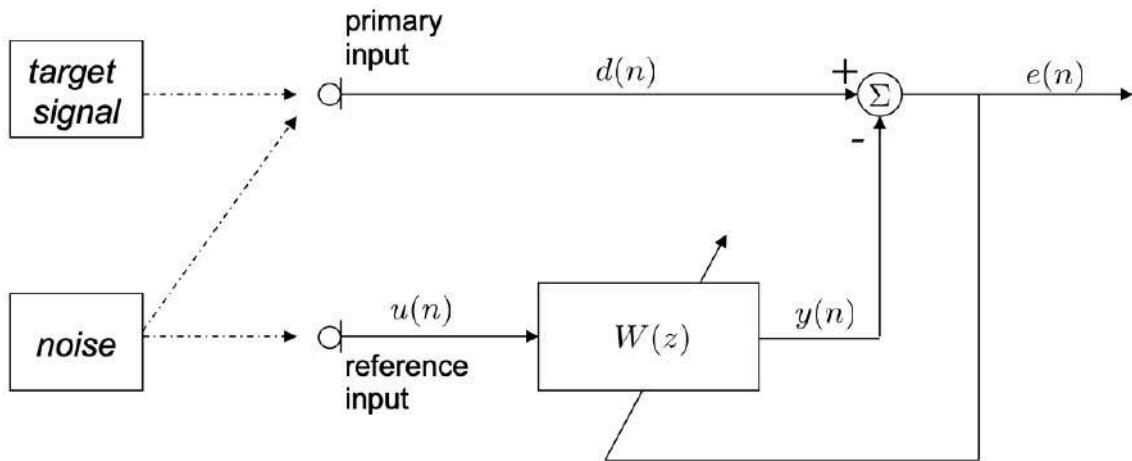
## Application for Adaptive Filtering

In today's lecture we will see four of the most common application for adaptive filtering. These are:

- 1) Noise Cancellation
- 2) Acoustic Echo Cancellation
- 3) Background noise Removal
- 4) Active noise control

### Noise Cancellation

It refers to situations where it is required to cancel an interfering signal/noise from a given signal, which is a mixture of the target signal and the interference. We can model the systems as follows.



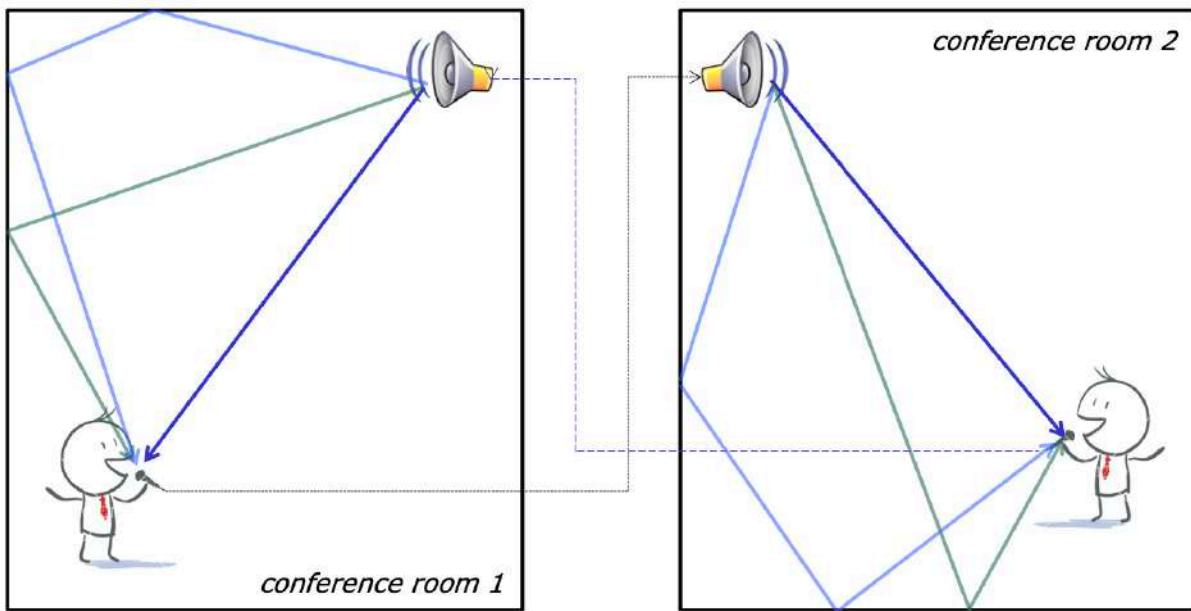
As we can notice, we need a reference input: a microphone placed very close to the signal we want to remove that allows the system to know which is the signal that has to be removed. It will be the input of the Wiener Filter. We could ask ourselves, is it mandatory to filter the noisy signal? Yes, it is! Even though the environment does not affect the signal in any way, we have to consider that the distance between the target microphone and the target signal source and the distance between the noisy source and the microphone that acquires it could not be the same. This means that we introduce a certain delay between the two lines that we have to catch up by means of the Wiener Filter.

Formally, we want to make the prediction error uncorrelated with the input of the system. This is the main principle that drives the estimation of the Wiener Filter. If we achieve this result, we are able to remove all the information related to the noise from the target signal. The very same scenario we saw some lessons ago. This is what is done, as an example, in Formula 1 cars where the driver needs to communicate with the team boxes as clearly as possible, without any interference from engine or mechanical parts of the car. The noise coming from the engine and mechanical parts is very high and we have to suppress it. To do so, we place a microphone where the noise originates, acquiring the reference noise signal  $u(n)$ . Then, we perform adaptive filtering of the signal  $u(n)$ , comparing the filter output with primary input  $d(n)$ , constituting the desired response. At this point, by making the difference between the two signals, we remove the noisy one. We can implement the LMS iteratively as follows:

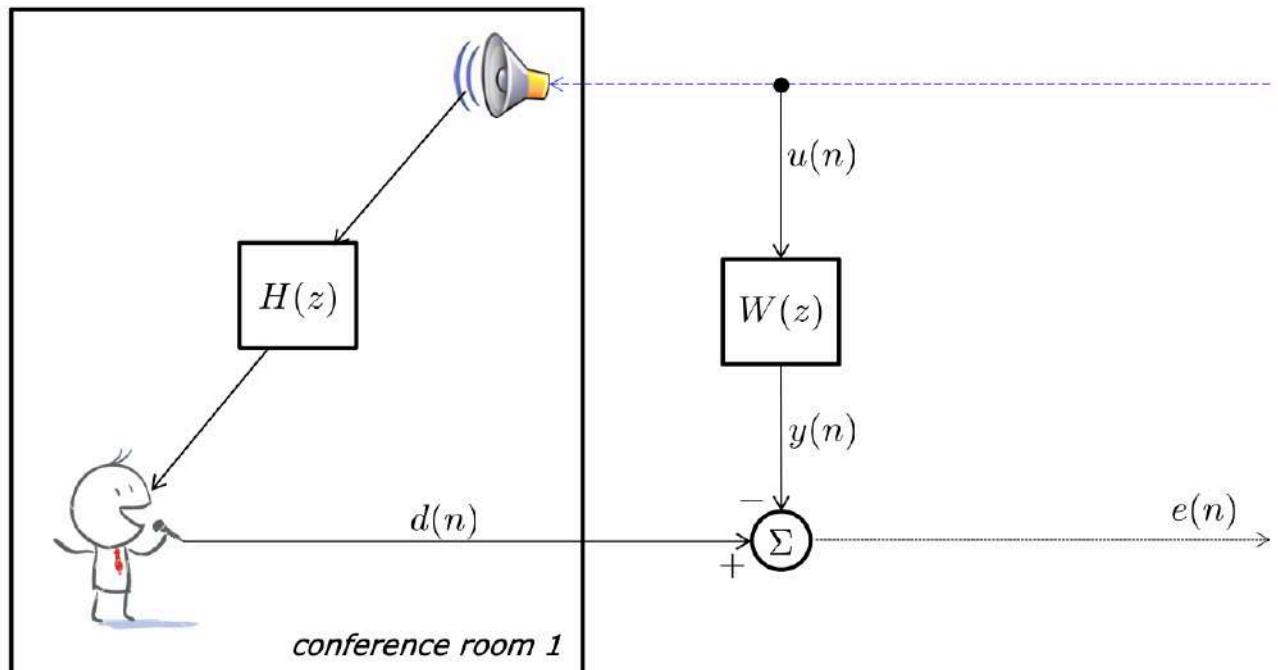
1. Filter the reference input:  $\underline{w}^T(n)\underline{u}(n)$
2. Estimate the target signal:  $e(n) = d(n) - y(n)$
3. Update the filter:  $\underline{w}(n+1) = \underline{w}(n) + \mu\underline{u}(n)e(n)$ . *Since most of the parameters that describe the noisy source changes in time, instead of having a one-shot update, we have an iterative estimation of the filter. It is updated considering the values of the filter at the previous step and making some adjustments with the estimated error.*

### Acoustic echo Cancellation

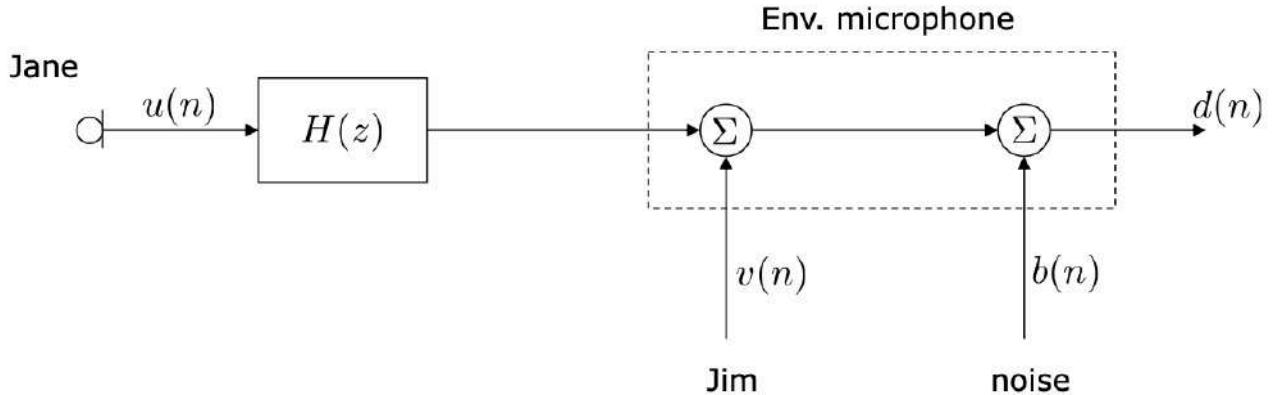
We started talking about Wiener Filter using this example. The situation is the following



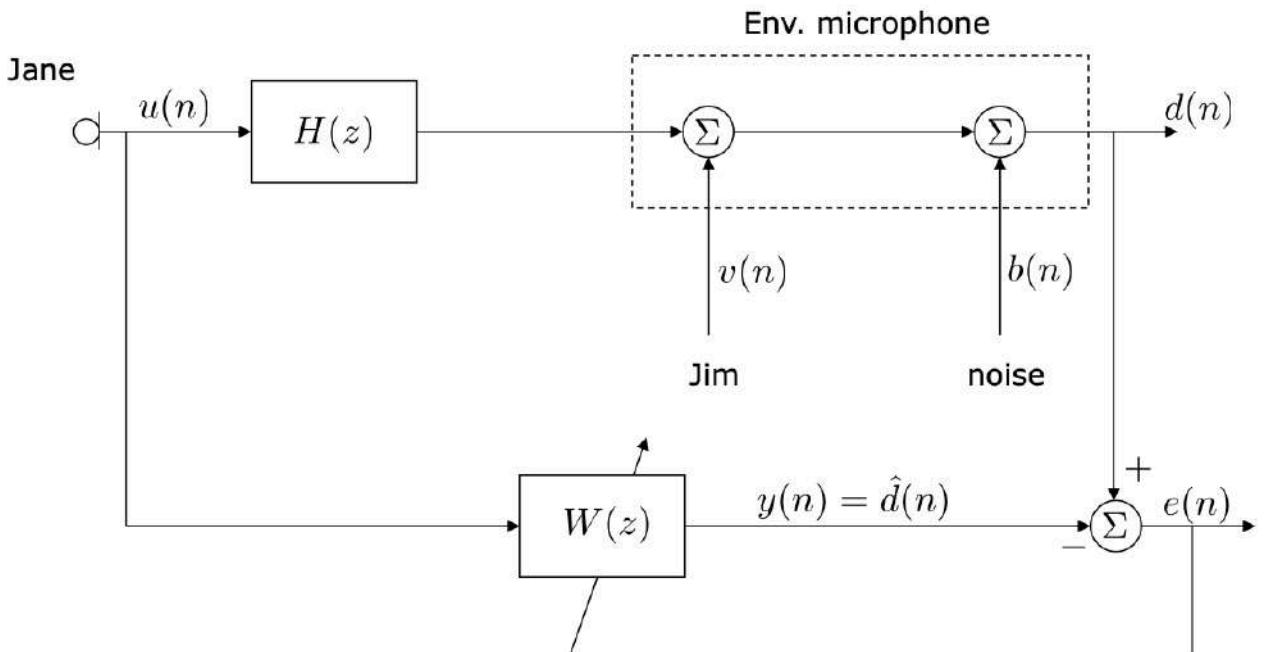
In a room there are one microphone open and one speaker open. We want that the voice that comes from the loudspeaker does not enter back in the microphone! Since it is almost impossible to break the loop acoustically, we let the microphone acquires both the signal and the echo and we remove the latter electronically. Here comes Wiener Filtering.



Another scenario in which acoustic echo cancellation is used is the one we face in remote class interaction. Let us describe it with an example: in a reverberant room two people (Jane and Jim, unfortunately not Jules et Jim or Jane and Michael) are speaking at the same time. Person 1 (Jane) is provided a wearable microphone. Another microphone acquires the signal from the environment. We want to acquire Jim's voice in order to make it hearable at home. The model of the situation is the following:



We need to remove Jane's voice from the signal to properly hear the Jim's speech. How can we do it?



We want to remove Jane's voice from the environmental microphone. We can use Jane's microphone to acquire the reference signal. By sending it to the Wiener filter, we obtain a signal by which removing the voice of Jane from the sound acquired by the environmental microphone. We must say that we can remove just the Jane's voice but we cannot do anything with all the contributions of the environmental noise since we do not have a proper reference signal to remove them. Furthermore, we have no information about Jim's voice (also in this case we have no reference signals), so we are not able to remove the eventual reverberation from its voice.

Let us formalise the concepts. Given the signal  $u(n)$  (acquired by Jane's microphone) and the desired signal  $d(n)$  (environment), we obtain  $y(n)$ , an estimation of  $d(n)$ .  $y(n)$  is estimated by the Wiener filter

$$y(n) = \underline{w}^T \underline{u}(n)$$

Filter that is driven by the residual error

The error signal is used as segregated signal every time it is only provided the signal to be cancelled.

$$e(n) = d(n) - y(n) = \underline{h}^T \underline{u}(n) + v(n) + b(n) - \underline{w}^T \underline{u}(n)$$

After an initial transient, the output signal converges to the desired signal, therefore the error signal contains “all the information in  $d(n)$  that cannot be estimated from the knowledge  $u(n)$  of and  $d(n)$ ”: the error signal statistically converges to  $v(n) + b(n)$ .

### *Echo Return Loss Enhancement (ERLE)*

In order to measure the effectiveness of an echo canceller, the Echo Return Loss Enhancement (measured in dB) is usually given. It measures the ratio between the energy of the residual echo after and before cancellation

$$ERLE|_{dB} = -10 \log_{10} \left( \frac{\sum_n [e(n) - v(n)]^2}{\sum_n [d(n)]^2} \right)$$

It is normalised to the energy of the signal (in order to make it signal independent).

If our signal is not stationary (that is our case since people speech and silence take turns) we have to consider that signals vary in time. In this case, also the ERLE varies in time. Since it is inconvenient to have a parameter of merit that varies in time, we measure instantaneous ERLE by averaging  $[e(n) - v(n)]^2$  and  $[d(n)]^2$  over  $\frac{1}{20}$  seconds long windows.




---

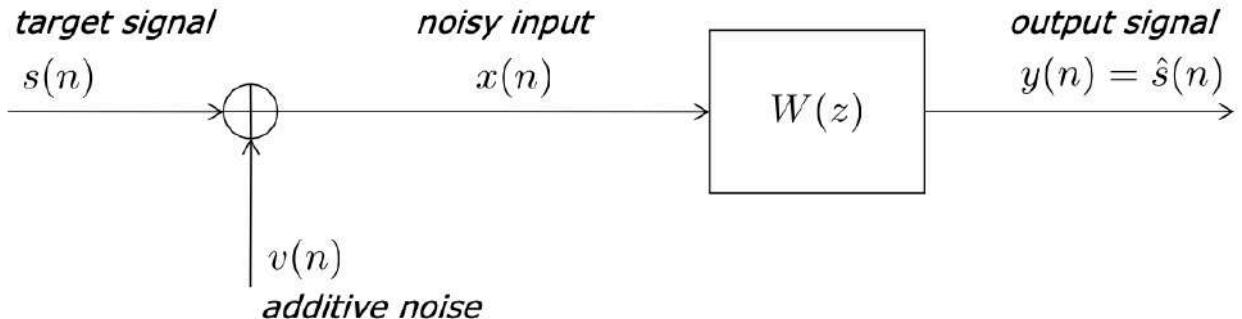
*Hissing noise*

---

Fruscio delle registrazioni

### Background noise removal

This is kind of restoration problem: we want to reduce hissing noise in an analogical recording (a tape). We remind that hissing noise is the disturb introduced by the material on which the information is stored.



We assume that:

- $s(n)$  and  $v(n)$  are independent;
- An estimate of the power spectrum  $P_v(\omega)$  of  $v(n)$  is available.

Our goal is to find the filter  $W(z)$  that minimises the MSE:

$$J = E\{|s(n) - y(n)|^2\}$$

The signal  $s(n)$  constitutes the desired response of the Wiener Filter. We know from the orthogonality principle that  $x(m)$  is decorrelated from the error.

$$E\{[s(n) - y(n)]x(m)\} = 0 \quad \forall n, \forall m$$

And the cross correlation between the desired signal and the noisy input signal is expressed by:

$$\begin{aligned}
 r_{sx}(n-m) &\stackrel{\substack{r_{sx}(\tau)=r_{sx}(-\tau)}}{=} E\{s(n)x(n-(n-m))\} = E\{s(n)x(m)\} \\
 &\stackrel{\substack{s(n)=w(n)*x(n)= \\ =\sum_k w(k)x(n-k)}}{=} \\
 E\left\{\sum_k w(k)x(n-k)x(m)\right\} &\stackrel{\substack{\text{Linearity of } E\{\cdot\} \\ w(k) \text{ is deterministic}}}{=} \sum_k w(k)E\{x(n-k)x(m)\} = \\
 &\stackrel{\substack{r_x(\tau)=E\{x(n)x(n+\tau)\} \\ \{n=n-k \\ n+\tau=m\} \\ \rightarrow -\tau=n-k-m \\ \rightarrow \tau=-(n-k-m)}}{=} \sum_k w(k)r_x(-(n-k-m)) \stackrel{\substack{r_{sx}(\tau)=r_{sx}(-\tau)}}{=} \sum_k w(k)r_x(n-k-m)
 \end{aligned}$$

That is exactly the Wiener-Hopf equation.

We know that we do not have access to  $s(n)$  since our desired signal is not available being corrupted by the noise. It is in fact the cleaned signal we want to obtain, but what we can read from the tape is only the corrupted signal.

How can solve the problem? By using the Wiener-Хинчин theorem

Reminder:

The Wiener-Хинчин theorem affirms that for a finite-power signal, the power spectrum  $S_x(f)$  is equal to the Fourier transform of the autocorrelation function.

$$S_x(f) = \mathcal{F}\{r_x(\tau)\}$$

In our case, since we have a signal that is called  $s(n)$ , we indicate the power spectrum with  $P_x(f)$

Therefor, the Fourier transform of the cross-correlation corresponds to the cross-spectrum

$$\mathcal{F}\{r_{sx}(n)\} = P_{sx}(\omega)$$

And the Fourier transform of the autocorrelation can be expressed as the power spectrum

$$\mathcal{F}\{r_x(n)\} = P_x(\omega)$$

In light of this, the Wiener-Hopf equations become:

$$\mathcal{F}\{r_{sx}(n-m)\} = \mathcal{F}\left\{\sum_k w(k)r_x(n-k-m)\right\} = P_{sx}(\omega) = W(\omega)P_x(\omega)$$

Since we know the relation between the power of the correlation (cross spectrum) and the power spectrum, we can find

$$W(\omega) = \frac{P_{sx}(\omega)}{P_x(\omega)}$$

As we said,  $s(n)$  and  $v(n)$  are independent. This means that they are also uncorrelated.

Reminder on uncorrelation and independence.

Two variables are independent if their reciprocal conditioning does not produce any effects.

- a) If  $X_1$  and  $X_2$  are independent, they are also uncorrelated ( $C_{x_1, x_2} = 0$ )
- b) If  $X_1$  and  $X_2$  are uncorrelated, we can say nothing about their independence.

If they are independent, given  $Z = X_1 + X_2$ , we have that

$$\sigma_Z^2 = \sigma_{X_1}^2 + \sigma_{X_2}^2 \quad r_{X_1, X_2} = \eta_{X_1} \eta_{X_2}$$

Furthermore, if  $\eta_{X_1} = \eta_{X_2} = 0 \Rightarrow r_{X_1, X_2} = C_{x_1, x_2} = 0$

It follows that

$$\begin{aligned} r_{sx}(n) &= E\{s(k)x(k+n)\} \underset{x(n)=s(n)+v(n)}{=} E\{s(k)[s(k+n) + v(k+n)]\} = \\ &= E\{s(k)s(k+n)\} + \underbrace{E\{s(k)v(k+n)\}}_{=0} = E\{s(k)s(k+n)\} = r_s(n) \end{aligned}$$

This in frequency domain corresponds to the fact that the cross-power of  $sx$  is equal to the power of  $s$

$$P_{sx}(\omega) = P_s(\omega)$$

Moreover, since all the cross terms are zero,

$$P_x(\omega) = P_s(\omega) + P_v(\omega) + P_{sv}(\omega) \underset{P_{sv}(\omega)=0}{=} P_s(\omega) + P_v(\omega)$$

At the end, if we plug these two definitions in the Wiener filter equation, we obtain that

$$W(\omega) = \frac{P_{sx}(\omega)}{P_x(\omega)} = \frac{P_s(\omega)}{P_s(\omega) + P_v(\omega)}$$

Looking at this ratio, we can observe that we need two values. One is  $P_s(\omega)$  and the other is  $P_v(\omega)$ . Since the noise is stationary (strong assumption but it works), we could find  $P_v(\omega)$  evaluating the noise in portions of the tape where the signal  $s(n)$  is zero (for example, during the pauses, or at the beginning, or at the end of the tape). We can estimate the power spectrum of the noise “on its own” there.

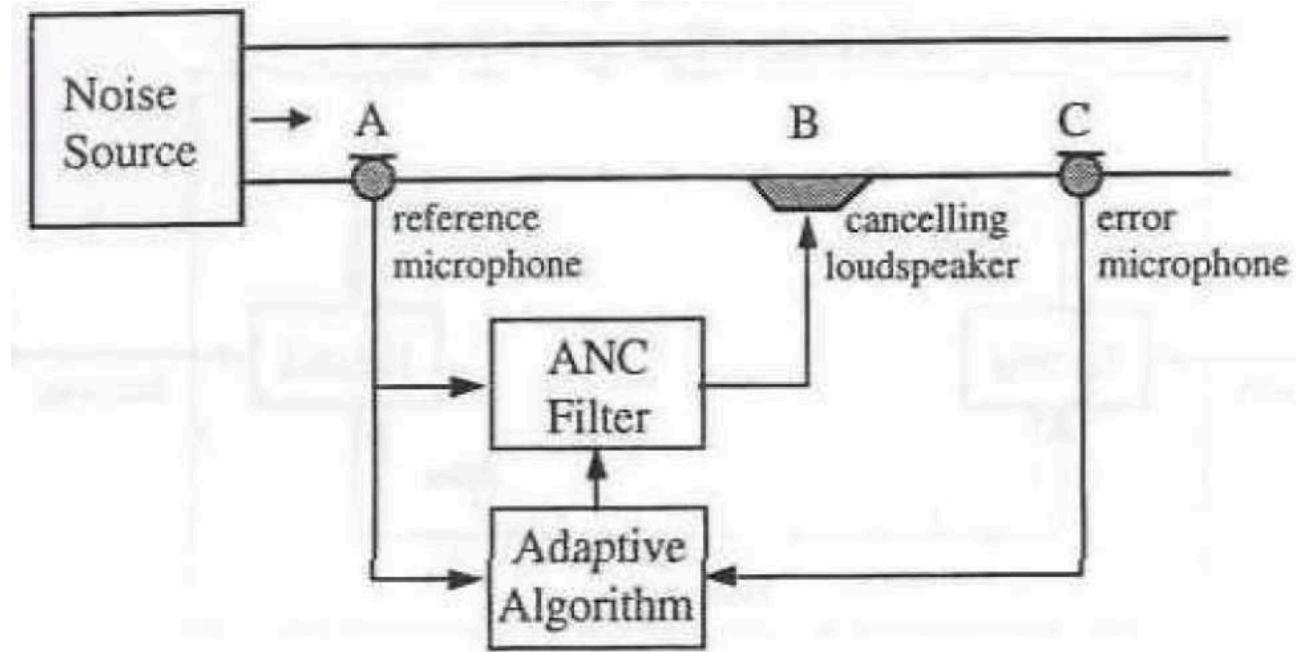
And about  $P_s$ ? We can exploit the fact that we have an estimate of  $P_v(\omega)$ , so we can extract  $P_s(\omega)$  by subtracting  $P_v(\omega)$  from  $P_x(\omega)$ .

At the end, our Wiener Filter will be obtained using these estimated values.

Let us now analyse the ratio.  $P_s(\omega)$  is divided by something that is always greater or equal to  $P_s(\omega)$ . This means that if  $P_s(\omega_o)$  is very small (the instrument is not playing, or the frequency played are out from the range of the recording filter)  $W(\omega_o)$  goes to zero. It implies that for that specific frequency only the noise will be present, so our filter will tend towards zero. This means that it will attenuate that frequency  $\omega_o$ . At the same time, if the signal energy is strong for that frequency band the filter tends towards 1, so the filter left the signal undistorted. This spectral attenuator is also sometimes called mask.

### Active noise control

This is slightly different from what we have seen before given the fact that is more related to acoustics. In the following example we explain how to cancel the noise from a fan system. We have a reference microphone that gives us a reference of the noise we want to cancel. The cancelling loudspeaker generates a waveform that is out of phase from the noise and that it is able to cancel it out. How can we do it? By using Wiener filter. The microphone gives an estimate of the performance of the cancelling and tells to the filter how to properly cancel noise.



## Features and Classification

Up to now we have seen just signal modelling processing. It is common nowadays to combine low-level signal processing tools with software techniques from research in artificial intelligence. Sometimes the classical signal processing models cannot perfectly analyse complex situations. The goal is to move beyond raw signal analysis to a deeper signal understanding.

AI find large application in:

- Automatic music transcription
- Speaker recognition systems: with signal modelling it is not easy to do it.
- Audio-based surveillance systems
- Musical genre classification
- Playlist generation/music recommendation

What we will see now is something that a few years ago could seem impressive. [Video]

This is a software to which we can ask the mood and that returns the songs. With signal modelling is very hard to do. Furthermore, there is also a browser that organises song in the emotional space. On one axis we have how strong an emotion is and on the other how much is positive or negative.

This is just to say how fast AI is developing.

We will only talk about a specific topic related to AI that is sound classification.

Given an audio track, sound classification consists in to associate a label or class to it, according to a possible set of labels or classes decided during the system definition.

By implementing a classificatory, we try to answer to these questions.

- Is this song rock or jazz?
- Is this a speech or musical signal?
- Is this a flute or a guitar?

Let us introduce the problem with a dummy example:

A forensic investigator wants to attribute a speech audio recording  $x_U(n)$  to a suspect using an automatic and objective technique. The candidate suspects are only two different people:

- Suspect A has the following voice characteristics
  - High loudness
  - High fundamental frequency
- Suspect B has the following voice characteristics
  - Low loudness
  - Low fundamental frequency



Why is it a classification problem? Because we want to find a label to pin on  $x_U(n)$ .

The investigator has M recordings  $x_1^A(n), x_2^A(n), \dots, x_M^A(n)$ , of the voice of suspect A and M recordings  $x_1^B(n), x_2^B(n), \dots, x_M^B(n)$ , from suspect B.

What we can do is to compare  $x_U$  with the other tracks and find if there are sort of similarities. This is what a human being do when they listen to the track. In terms of signals, we have to come up with a way of measuring if there are similarities. We do not want to compare the two signals sample by sample: it does not make any sense! Two signals perfectly match sample by sample if and only if they are the same signal. We have to extract some features from the signals to compare them.

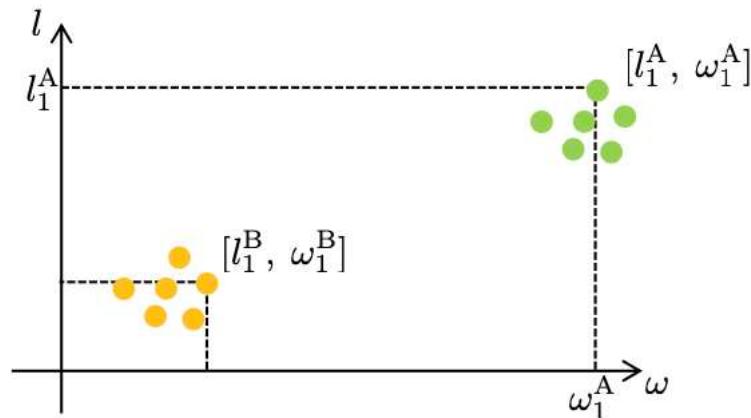
### Feature extraction

The investigator computes for each audio recording  $x(n)$  two quantities

- $l$ : an approximation of the loudness as the average signal amplitude
- $\omega$ : the fundamental frequency using LPC residual computation and peak finding

At the end, we will have two numbers associated to all the tracks. They will be strongly correlated to the pieces of information that will be helpful to solve the problem.

Given the fact that we have two pieces of information, we can represent the data tracks as points on a plane whose coordinates will be given by their loudness and their frequency. We have in green the tracks related to the test A and in yellow the one related to test B.



Now, in order to understand if the unknown track most likely belongs to suspect *A* or suspect *B*, we draw also the point (in red) related to the track of the suspects. Let us see what happens considering different scenarios

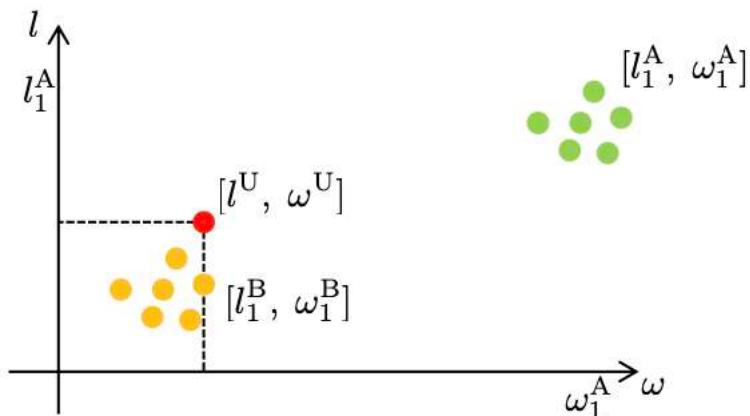
During lecture we approached to the problem interactively using the test at the following link  
<https://socrative.com/>

Login as student

ID: BESTAGINI

We report there just the solutions.

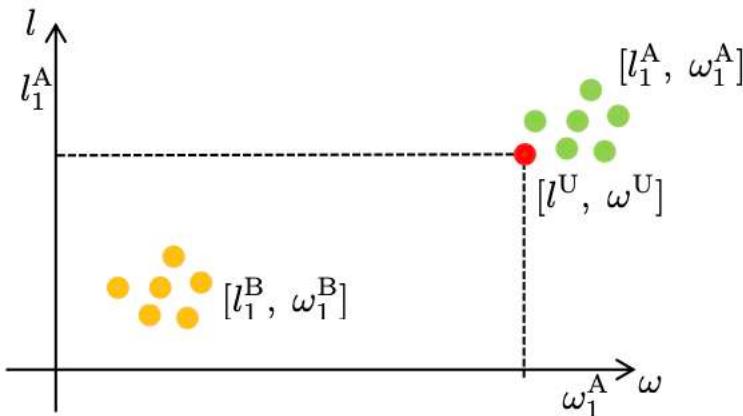
### Test (Case 1)



The investigator concludes that  $x_U(n)$  likely belongs to suspect B since the red point is very close to the orange ones.

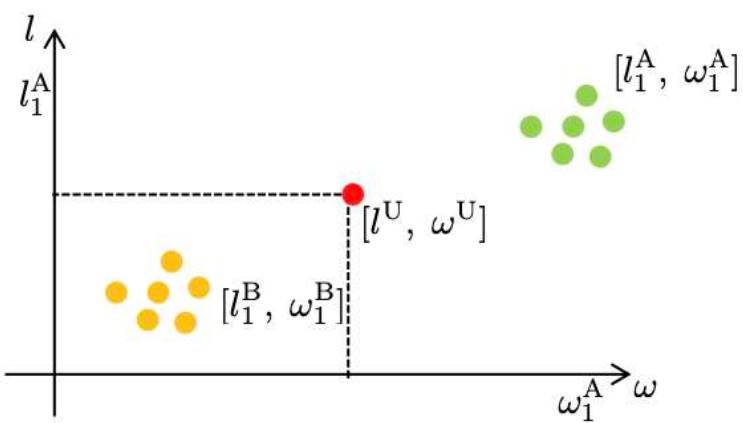
We are using a lot of conditional! In principle, we do not actually know if the suspect and the test B are the same person. We are just testing the similarities!

### Test (Case 2)



The investigator concludes that  $x_U(n)$  likely belongs to suspect  $A$ , since the red point is very close to the green ones this time. Close, but not amongst them or superimposed to one of them.

### Test (Case 3)

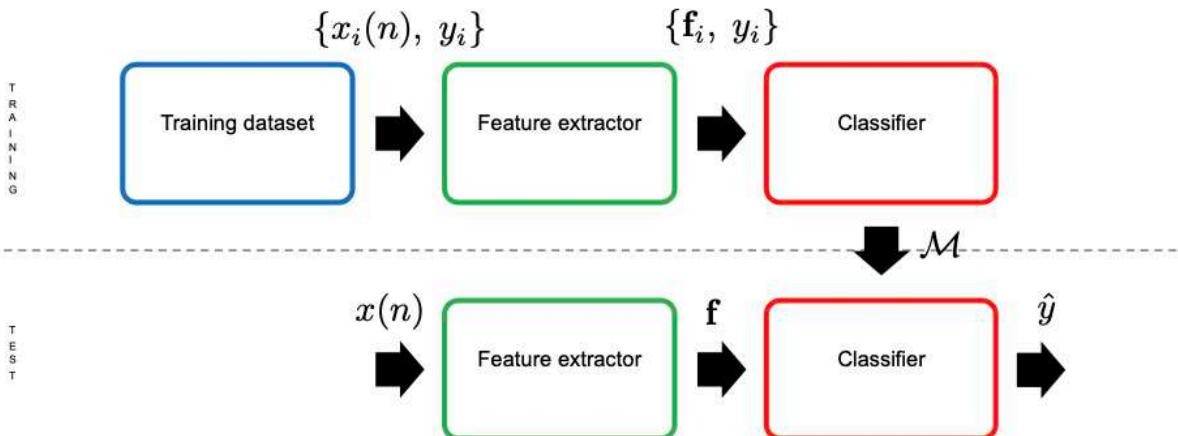


The investigator concludes that  $x_U(n)$  might or might not belong to any suspect. In fact, we notice that the red point is equidistant from orange and green points.

Why are we so cautious in our verdicts? Because we are missing a lot of information: we are guessing something basing our enquiry on the way the problem has been set. Maybe,  $x_U(n)$  belongs to someone else of whom we do not possess the tracks.

This exercise wants to warn us that we have to pay attention to the hypothesis we are making when we work with Machine Learning. Furthermore, that the results are always based on statistics and that the validity of a statistic increases as the amount of data on which it is evaluated increases. Maybe, if we had more information related to the yellow points, the centroid of them would be positioned elsewhere.

Let us now introduce some formalisms to properly describe the classification problem.

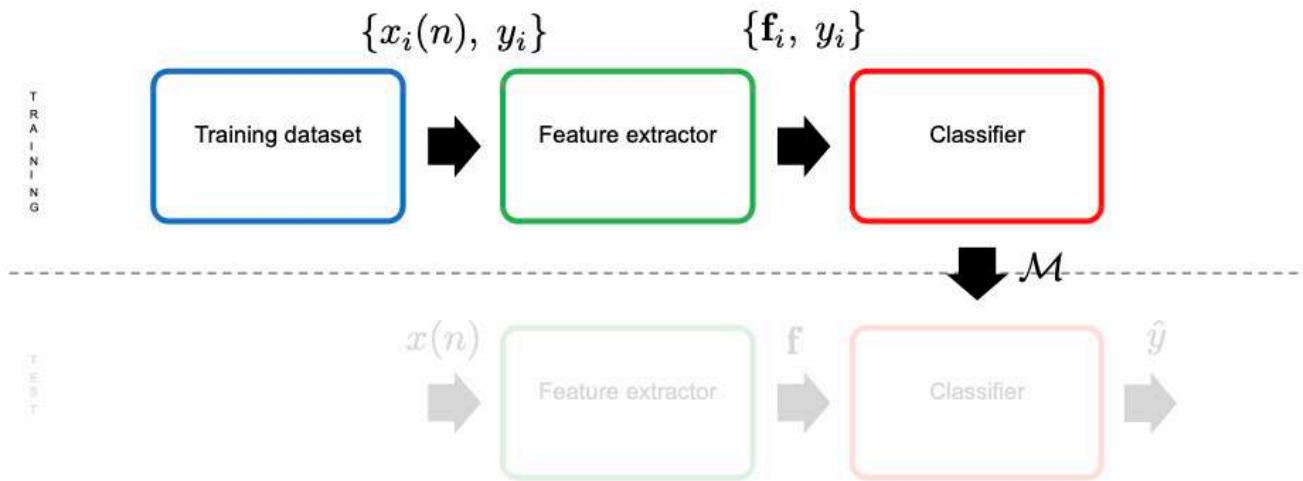


This is the simplest pipeline we can use. We have a training dataset  $\{x_i(n), y_i\}$  where all the data-tracks  $x_i(n)$  have a label  $y_i$ .

The feature extractor takes as input an audio track and returns a vector with the information we need to make a proper classification. Formally, it maps audio signals into vectors of fixed length  $\underline{f} = [f_1, f_2, \dots, f_N]$ . Last block we have here is the classifier. The AI system that maps feature vector to labels.

How do we use it?

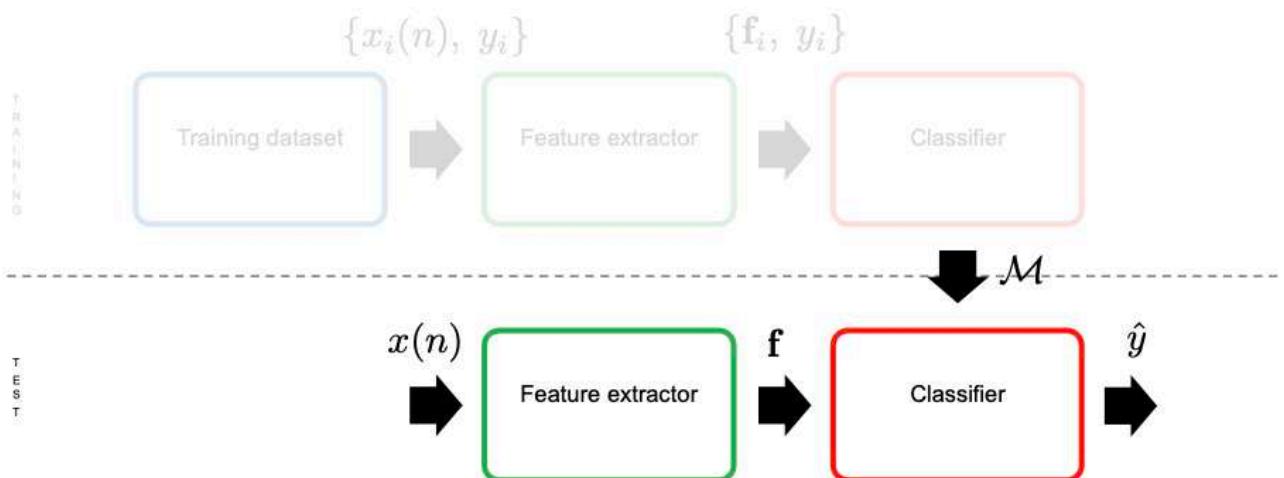
Firstly, we have to train it. This process is more or less what we were doing when we were visualising the features of A and B on the plane before.



The training steps are the following:

- 1) Features are extracted for all tracks in the training dataset.
- 2) The classifier is fed with  $\{\text{feature vector}, \text{label}\}$  pairs and “learns” a mapping between features and labels.
- 3) The trained model  $\mathcal{M}$  is saved. The classifier learns which is the mapping. Graphically, it splits the plane in regions and dedicate each region to every class.

At this point we have a well-trained system. Ready to classify new songs.



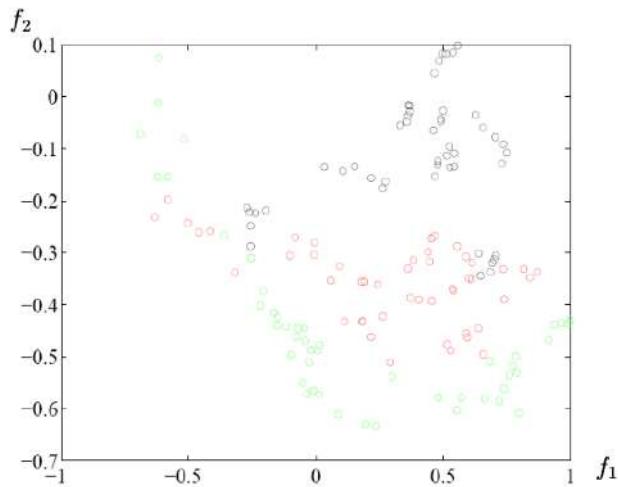
We go on with the test steps:

- 1) Features are extracted from the test track  $x(n)$
- 2) The trained classifier is fed with the feature vector  $\underline{f}$
- 3) The classifier estimates a candidate label  $\hat{y}$  to associate to  $x(n)$

### Geometric interpretation

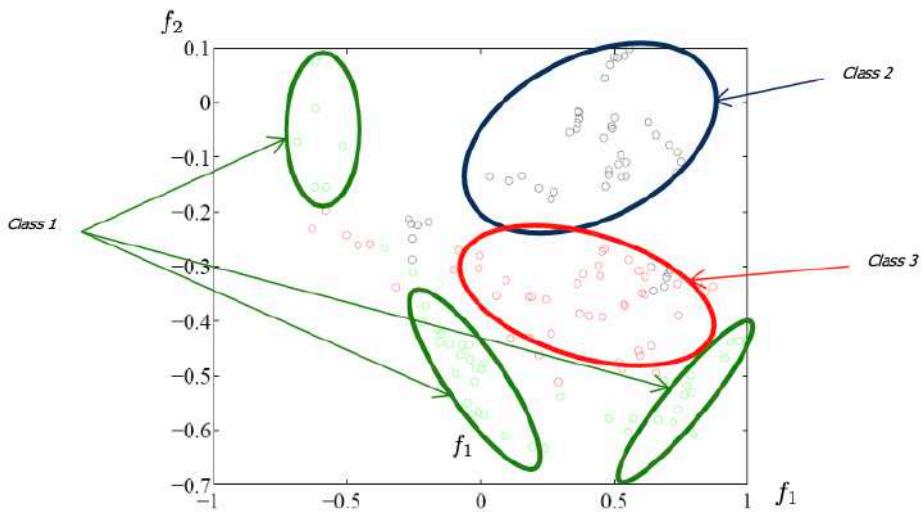
Let us see which is the Geometric Interpretation of the classification process.

As we saw in the dummy example, we can represent the N-element feature vector related to a track as a point in a N-dimensional space



The classifier does nothing but dividing the space in regions. How so?

If the chosen features are relevant for the problem, the points in the feature space cluster so that each cluster in the feature space can be assigned to an audio class. The classifier learns how to bound regions according to their classes.



### Some final considerations

In order to get a successful classification, we have to keep an eye on some crucial points:

- **Choice of a set of relevant features:** clusters of different classes need to be distinct (i.e. non-overlapping) and points within each cluster need to be close to the centroid of the cluster. This fact is strongly affected by the choice of the features. The features extractor can be based on signal processing or convolutional neural network. Sometimes it is merged with the classifier.
- **Choice of a suitable classification system** that performs partitioning of the feature space. Depending on the problem, we will need more or less fine system of classification. For instance, we are not simply separate with a straight line the elements in the second example, we are using ellipsis increasing the complexity of the classifier.
- **Choice of a relevant training set:** a choice of a training set that is not representative of the problem severely impairs the classification ability. But not too much, otherwise we bump into overfitting.

## Audio Features

Let us see a set of audio features that are mostly used nowadays. Audio features are also called descriptors. We can divide the descriptors into different families.

- 1) **Basic descriptors:** features related to audio statistics;
- 2) **Harmonicity descriptors:** features capturing complex audio information related to the harmonic components of an audio track;
- 3) **Timbral descriptors:** features capturing timbre information (e.g., for instrument or speaker recognition);
- 4) **Other descriptors:** depending on the application, different kind of hand-crafted formulas can help extracting audio characteristics.

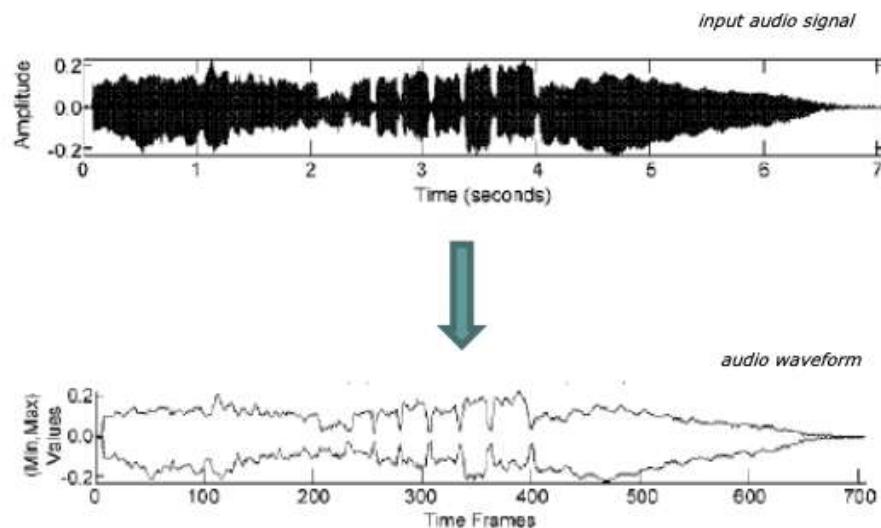
A little resume of the nomenclature to be able to surf into the equation we are going to use:

$t$	index of time samples
$s(t)$	input digital audio signal
$F_s$	sampling rate
$l$	index of the time frame
$s_l(t)$	$l$ -th frame from $s(t)$
$N_{hop}$	number of samples between successive frames
$l_w$	length of a timeframe [sec]
$N_w$	number of samples in each time frame
$L$	number of windows in $s(t)$
$k$	index of the frequency bin
$S_l(k)$	$k$ -th coefficient of the DFT of $s_l(t)$
$P_l(k)$	$k$ -th coefficient of the power spectrum of $s_l(t)$
$N_{ft}$	number of bins of the DFT/power spectrum of $s_l(t)$ )

## Basic descriptors

### 1. Audio Wave From (AWF)

It is defined as a set of two values (minRange, maxRange) that vary in time. The former represents the lower limit of audio amplitude in the frame, while the latter the upper one.

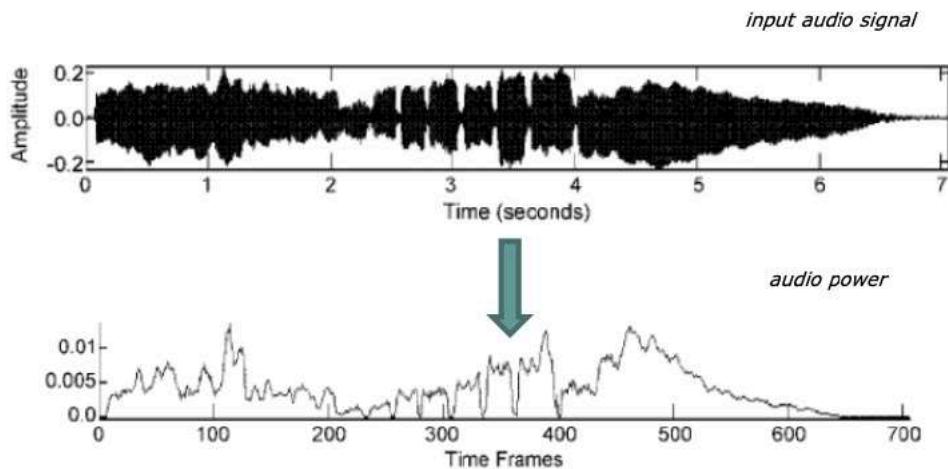


Sometimes we do not want to associate a label to each frame, what it is typically done is to compute the AWF frame by frame and, at the end, extract some statistical info from the AWF

## 2. Audio Power

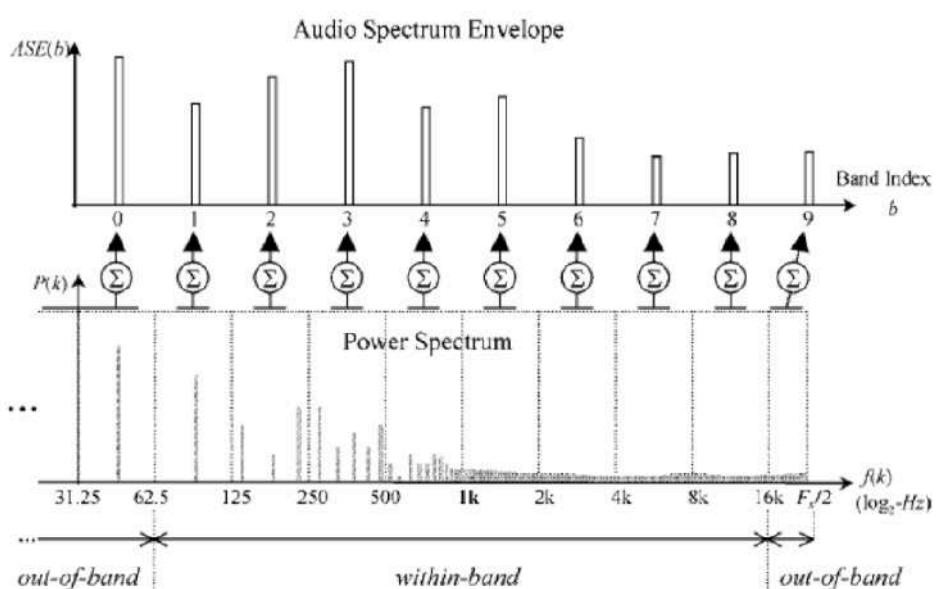
The audio power describes the temporally smoothed instantaneous power of the audio signal. The AP coefficients are the average square waveform values  $s(t)$  within successive non-overlapping frames ( $N_{hop} = N_w$ ).

$$AP(l) = \frac{1}{N_{hop}} \sum_{t=0}^{N_{hop}-1} |s(t + lN_{hop})|^2, \quad 0 \leq l \leq L - 1$$



## 3. Audio Spectral Envelope

It is a sort of reduced - summarised - version of the spectrum. It is easier to explain it with a picture.



It is obtained by summing the energy of the original power spectrum within a series of frequency bands. The bands are logarithmically distributed (based on logarithms) between two frequency edges. ( $loEdge$ ,  $hiEdge$ ).

The spectral resolution  $r$  of frequency bands within ( $loEdge$ ,  $hiEdge$ ) interval can be chosen from eight possible values ranging from  $\frac{1}{16}$  of an octave to 8 octaves. In the following formula, we denote with  $b$  the index of the band in which ASE is computed.

$loF_b$  and  $hiF_b$  are expressed in Hertz, while  $loK_b$  and  $hiK_b$  are the number that identifies  $loF_b$  and  $hiF_b$  with respect to the fundamental frequency.

$$ASE_b(l) = \sum_{k=loK_b}^{hiK_b} P_l(k), \quad 1 \leq b \leq B$$

Where the limits of each band are:  
 $loF_b = loEdge \cdot 2^{(b-1)r}$   
 $hiF_b = loEdge \cdot 2^{br}$

#### 4. Spectrum Centroid

Nomina sunt consequentia rerum, it is the centre of mass.

All coefficients below a certain frequency  $K_{low} = 62.5$  Hz are summed and represented by a single coefficient, in order to prevent a non-zero DC component and/or very low frequency components from having a disproportionate weight.

Let us see how to evaluate it. First, we have to define a modified spectrum.

$$P'(k) = \begin{cases} \sum_{k=0}^{K_{low}} P(k), & k' = 0 \\ P(k + K_{low}), & 1 \leq k' \leq \frac{N_{ft}}{2} - K_{low} \end{cases}$$

Since we condensed the spikes for  $k \leq K_{low}$  into a single one, we have to shift all the frequencies in order to make them correspond to their spike again.

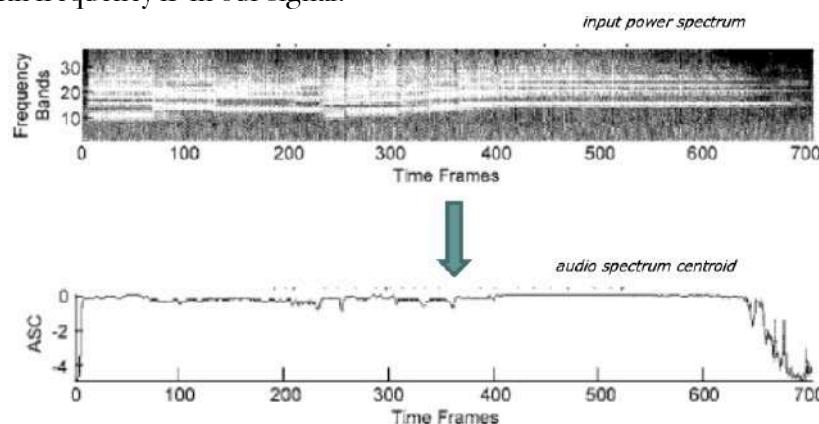
$$f'(k') = \begin{cases} 31,25 \text{ Hz}, & k' = 0 \\ f(k + K_{low}), & 1 \leq k' \leq \frac{N_{ft}}{2} - K_{low} \end{cases}$$

Simply speaking, we sum all the power coefficients in a single power coefficients, and we append the next ones to it. Given this mapping between frequency, we define the spectrum centroid by applying the definition of centroid (but logarithmically). With  $\frac{1}{1000}$  we just scale the frequency to Khz and we apply logarithm to obtain a logaritmical representation.

$$ASC(l) = \frac{\sum_{k'=0}^{\frac{N_{ft}}{2}-K_{low}} \log_2 \frac{f'(k')}{1000} P'_l(k')}{\sum_{k'=0}^{\frac{N_{ft}}{2}-K_{low}} P'_l(k')}$$

This formula can be interpreted in two ways:

1. The definition of the centroid: weighted average of the frequency on how much energy there is frequency by frequency.
2. The percentage of power that we observe in  $k'$  that can be interpreted as the probability of having something with frequency  $k'$  in our signal.




---

	<i>Centre of mass</i>	
	Baricentro	

---

##### 5. Audio Spectrum Spread

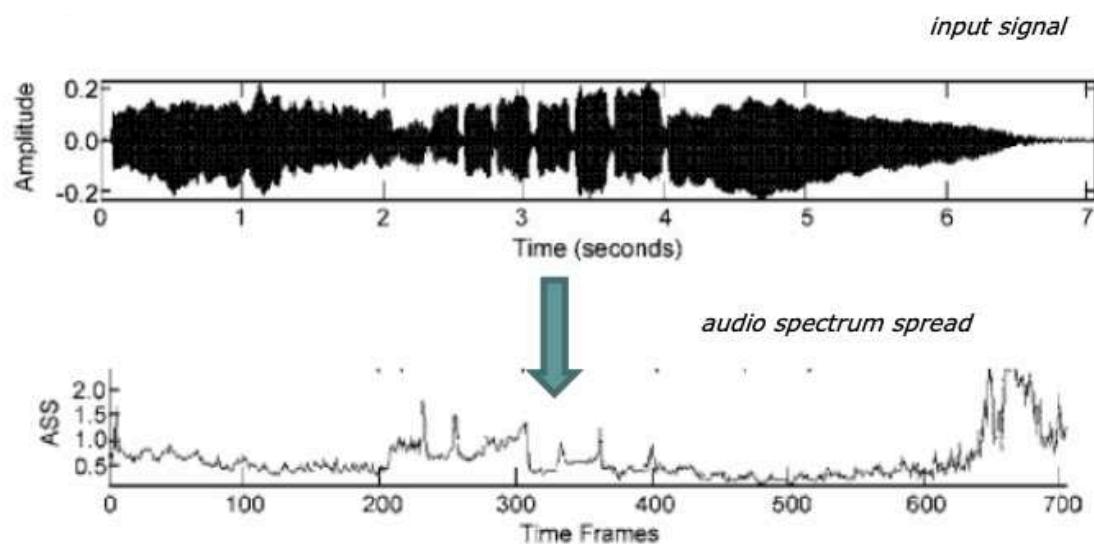
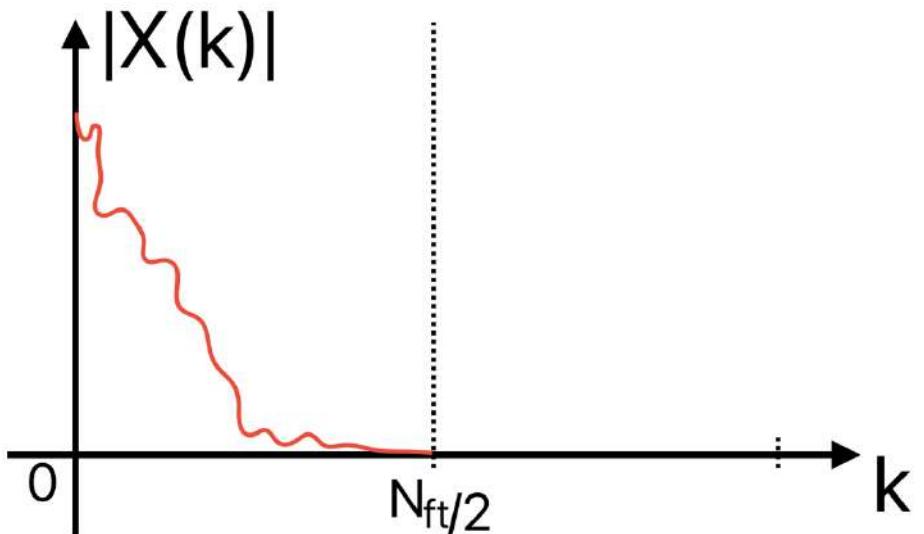
The spectrum spread can be linked to the concept of standard deviation. It returns a measure of how much the spectrum is spread around the centroid.

Therefore, the ASS feature is extracted by taking the root-mean-square (RMS) deviation of the spectrum from its centroid ASC.

$$ASS(l) = \sqrt{\frac{\sum_{k'=0}^{\frac{N_{ft}}{2}-K_{low}} \left[ \log_2 \left( \frac{f'(k')}{1000} - ASC(l) \right) \right]^2 P'_i(k')}{\sum_{k'=0}^{\frac{N_{ft}}{2}-K_{low}} P'_i(k')}}$$

We notice that  $N_{ft}$  is divided by two. This means that the negative frequencies are not considered. This is typical in signal processing function (for example the Librosa's ones) given the fact that the negative frequencies are useless in the audio field. Being the signal always real, they are just a copy of the positive frequencies.

As we can see in the picture on the right, we neglect what we have after  $\frac{N_{ft}}{2}$ . We also remind that, being the DTFT periodic, what we have on the right is just a repetition of the period between  $-\frac{N_{ft}}{2}$  and 0. It can be seen also as the specular version of what we have from 0 to  $\frac{N_{ft}}{2}$ . As we said, since they are nothing but repetitions, we can neglect them.



## Harmonicity Descriptor

The above-mentioned basic spectral give a smoothed representation of power spectra. They cannot reflect the detailed structure of periodic sounds because of a lack of frequency resolution. The audio harmonicity descriptor provides different measures of the harmonic properties of a spectrum.

### 1. Harmonic Ratio

The harmonic ratio is nothing but the maximum of the normalisation of the autocorrelation of the signal.

If the mean value of the signal is zero. It corresponds to the maximum of the coefficient of correlation, since the autocorrelation corresponds to the autocovariance.

Formally, given the normalised autocorrelation function

$$\Gamma_l(m) = \frac{\sum_{t=0}^{N_w-1} s_l(t)s_l(t-m)}{\sqrt{\sum_{t=0}^{N_w-1} s_l^2(t)} \sqrt{\sum_{t=0}^{N_w-1} s_l^2(t-m)}}, \quad 1 \leq m \leq M, \quad 0 \leq l \leq L-1$$

The harmonic ratio is defined as the maximum of the normalised autocorrelation function in the time interval  $\mathcal{I}_m = [M_{min}, M_{max}]$  where:

- $M_{min} > 0$ . We do not consider the case  $m = 0$  since, in that case, we would evaluate the autocorrelation two signals are perfectly superimposed and it is obvious that, in that case, we have the maximum value.
- $M_{max} = T_{max}F_s$ . We remind that if the signal is periodic the autocorrelation is periodic as well. So, the larger is this maximum ( $T_{max}$  maximum fundamental period that can be estimated), the more harmonic is the signal.

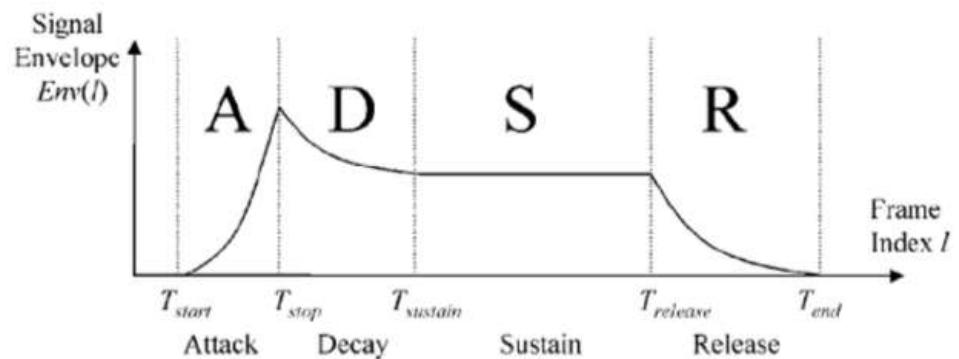
The final definition of the harmonic distortion is

$$HR(l) = \max_{m \in \mathcal{I}_m} \Gamma_l(m)$$

## Timbral Descriptors

The timbre is the feature of the sound that allow one to distinguish two sounds that are equal in loudness, pitch and subjective duration. One of the things that characterises the timbre is the envelop. By an electronical engineering, it can be seen as the low pass version of the rectified waveform. Simply speaking, it is the shape that we

obtain if we interpolate all the peaks of the signal in time. The signal envelope describes the energy change of the signal and is generally equivalent to the so-called ADSR (Attack, Decay, Sustain, Release) of a musical sound.



- *Log attack time*

The log attack time is defined as the time sound envelope takes to reach the maximum amplitude of a signal from a minimum threshold time. Its main motivation is the description of the onsets of single sound samples from different musical instruments. LAT is defined as the logarithm of the duration from time  $T_{start}$  when the signal starts to time  $T_{stop}$  when it reaches its maximum value (for a percussive sound) or its sustained part (for a sustained sound, a sound that has no decay phase).

$$LAT^P = \log_{10}[T_{stop} - T_{start}]$$

$$LAT^S = \log_{10}[T_{sustain} - T_{start}]$$

- Mel-Frequency Cepstrum Coefficients

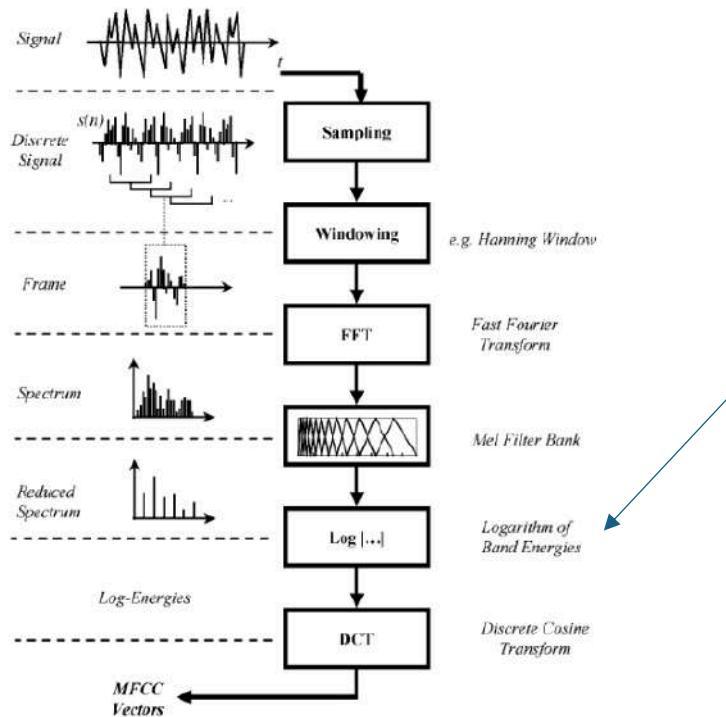
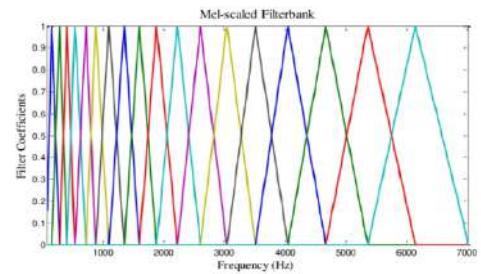
MFCC is one of the most used when we want to classify human voice and musical signals. In particular, the MFCC parametrization of speech has proven to be beneficial for speech recognition. A non-linear Mel-frequency scale is used, which approximates the behaviour of the auditory system. To convert a frequency in Hertz into its equivalent in Mel, the following formula is used:

$$f_{Mel} = 1127.0148 \cdot \log\left(1 + \frac{f_{Hz}}{700}\right)$$

The MFCCs are based on the extraction of the signal energy within critical frequency bands by means of a series of triangular filters whose centre frequencies are equally spaced according to the Mel scale.

The log-energy of the spectrum is measured within the pass-band of each filter, resulting in a reduced representation of the spectrum. The Cepstral coefficients are finally obtained through a Discrete Cosine Transform (DCT) of the reduced log-energy spectrum.

So, how can we compute them? By following the pipeline



At this point the magic happens, we apply log to the amplitude and the DCT to the spectrum. We are slightly doing a FFT twice. The output of this operation is the so-called CEPSTRUM (fun fact: it comes from the anagram of Spectrum)

## Other descriptors

### 1. Zero crossing-rate

It is computed counting the number of times that the audio waveform crosses the zero axis.

$$ZCR(l) = \frac{F_s}{2N_w} \sum_{t=1}^{N_w-1} |sign[s_l(t)] - sign[s_l(t-1)]|$$

We are comparing the sign of one sample of the signal and the one of the next sample of the signal. If one is positive and the other is negative, it means that there is a zero-crossing, and the difference becomes two. Hence, we have the sum of all the times that the signal passes to zero times two (under the assumption that we have a sample that is never a real zero (highly probable in real systems)). Then, we scale it by  $F_s$  and  $N_w$  (fraction of zero crossing in percentage of samples) and at the end we divide it by two to remove the product by two that we get from the difference of the signs.

### 2. Spectral roll-off

It tells us which is the frequency below which we have the 85% of the accumulated spectrum magnitude.

$$\sum_{k=0}^{k_{roll}} |S_l(k)| = 0.85 \sum_{k=0}^{\frac{N_{ft}}{2}} |S_l(k)|$$

### 3. Spectral Flux

It is defined as the average variation of the signal amplitude between adjacent frames. It is computed as the average squared difference between two successive spectral distributions:

$$SF = \frac{1}{LN_{ft}} \sum_{l=2}^L \sum_{k=1}^{N_{ft}} \{\log[|S_l(k)| + \varepsilon] - \log[|S_{l-1}(k)| + \varepsilon]\}^2$$

We add the  $\varepsilon$  to correct the case in which we have a sample equal to zero. Without it, when the signal is equal to zero, we would obtain as a result of the logarithm  $-\infty$  and we would make the sum diverge.

## Classification methods

We know how to extract the descriptors. We have now to go through the classifier methods. The classifier we will use are the **supervised classifiers**. The name comes from the fact that it “learns” a mapping between input data and labels. We will introduce only very simple classifiers.

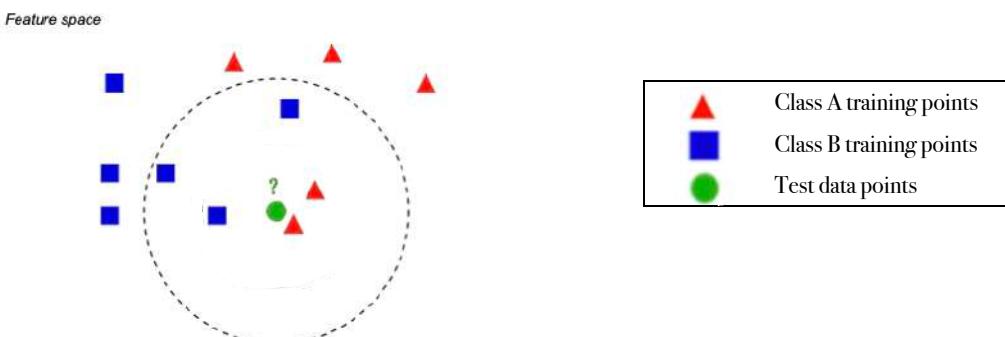
### K Nearest Neighbour KNN

The simplest one. It simply stores all the training data in a huge table with the features associated to a specific label. If we think about the example we did at the beginning of the lesson, the table should be like the one to the right. Given the data set, how do we perform the classification? Let us represent all the training elements in the feature space (the red triangles and the blue squares). Let us now represent also the element that has to be tested (the green point). We choose a value of  $K$  by which to create a circle around the green point. Then, we ask to ourselves how many red triangles and blue square are inside the circle. If we have more red elements than blue one, we classify the green dot in the red class. Otherwise, if the number of blue elements is larger than the red one, we classify the testing element in the blue class. With this value of  $K$ , there are more red elements than blue one inside the circle, so green is mapped into red.

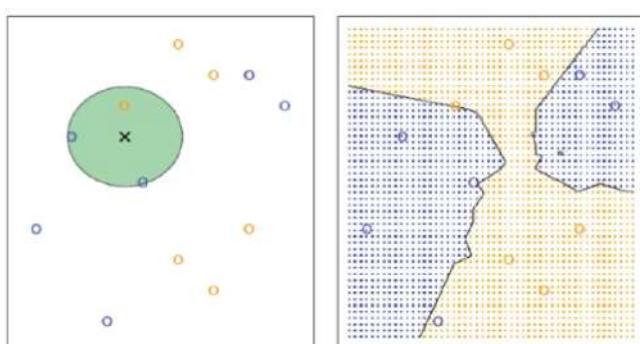
$l$	$\omega$	$y$
$l_1^A$	$\omega_1^A$	$A$
$l_2^A$	$\omega_2^A$	$A$
$\vdots$	$\vdots$	$\vdots$
$l_5^B$	$\omega_5^B$	$B$



We suppose now to increase the value of  $K$ .



What we notice is that, now, there are more blue elements inside the circle than red ones.



Depending on  $K$ , the result highly changes. Furthermore, the higher, the  $K$  the duller is the classifier: it becomes not more able to distinguish different cases. In the extreme case of  $K \rightarrow \infty$ , the green point is always classified as a blue one regardless of its position, since the circle contains all the feature space and, inside it, there are more blue dots than red ones. We can also visualise which is the evolution of the classes in the feature space.

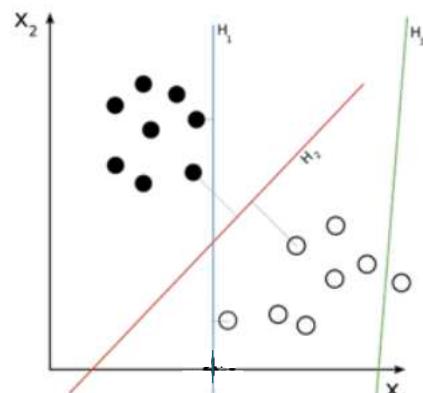
Pros	Cons
KNN has “zero” training time	KNN has problems in high dimensional spaces, which require approximate methods for neighbours finding
Cost of finding the closest K neighbours can be reduced at run time through indexing	Since we do not store anything in memory, the cost comes at execution time: there are lot of operation to be computed for the comparation.
We do not store anything in memory.	

### Linear Classifier

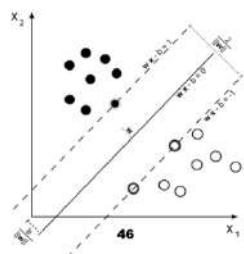
The linear classifier simply divides the space using straight lines. If we are in N-dimensional space, we divide it in regions by means of an n-hyperplane. As we can notice by looking at the example on the right, we can choose different lines to create different region in a same plane. The red one and the blue one seem a choice more appropriate than the green one.

To train a linear classifier we estimate a line separating training points belonging to the two classes using iterative algorithms. To test a new data, we check whether the new data point to classify lies on one side or the other side of the separating line / plane/ hyper-plane.

The most famous linear classifier is the



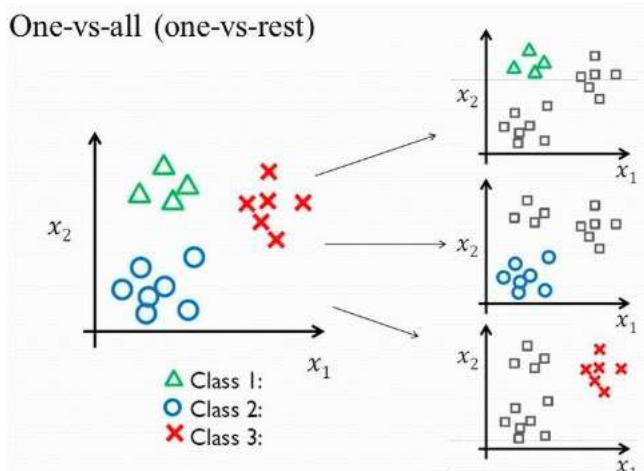
### Linear Support Vector Machine (SVM)



It introduces the concept of margin: the distance between the closest element of a class to the separation line and the separation line itself. We want to maximise this margin. It is substantially an optimisation problem.

All the examples that we have seen up to now are binary classification methods. Sometimes we have multiple classes. In these cases, we simply have to adapt the binary model to work with them.

#### One-vs-all (one-vs-rest)



One simple kind of adapting technique is the one-vs-all. The line will be traced one class (green, red or blue) and the other remaining elements of the other two classes.

#### How does the test work?

When a new data arrives, we compare it with the three classes, and we try to merge the results.

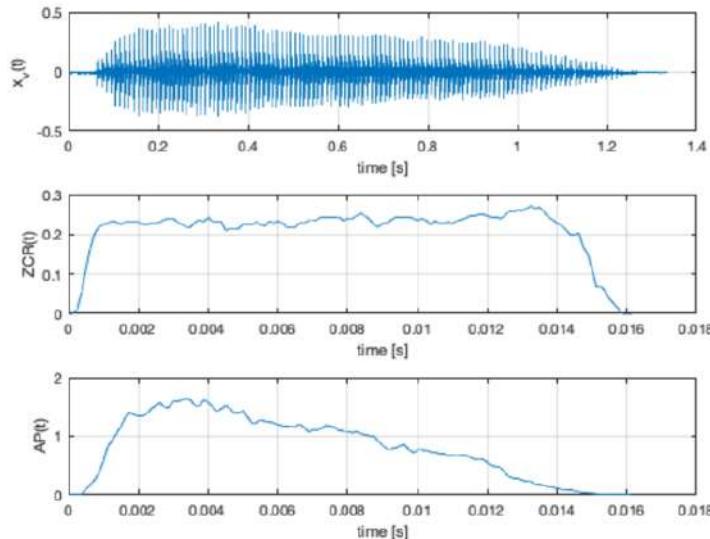
## Feature extraction and classification on MATLAB

In today's lecture we will try to perform feature extraction and classification on MATLAB.

### Exercise 1 - Feature Extraction

- Load the file 'voiced.wav'
- Define a Hamming window of length 40 ms
- Extract ZCR and AP from each window of the audio signal, using hop size of 10 ms
  - o Extract a 40 ms audio excerpt
  - o Apply Hamming window
  - o Compute ZCR
  - o Compute AP
  - o Save feature vector [ZCR, AP]
  - o Repeat for all windows...
- Plot the following figures
  - o The audio waveform
  - o The ZCR in time
  - o The AP in time

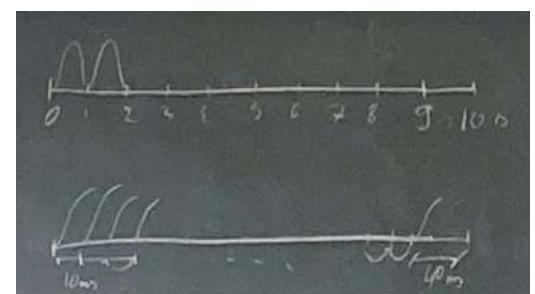
The result we should obtain is the following:



*Note on windowing:*

*First of all, given a signal that lasts  $N$  samples, how many windows do we need to crop it completely if the hop-size is 10 ms and the window length is 40 ms?*

*We know that we move the window every 10 ms. Therefore, we have to count one window each 10 ms. But, at the end we will bump into an issue: the last window should be long at least the window length so when we are at 40 ms before the end, the signal cannot be divided in portion of 10 ms anymore. What we do is simply subtracting 40 ms to the length of the signal and then dividing the result by 10 ms.*



Note on the mismatch of ranges:

The acoustic power is in the range of  $10^{-3}$  while the zero-crossing rate is in the range of 2000. For visualisation purpose, it is very difficult to see them on a same graph. Furthermore, if we measure the norm 2 having as coordinates two numbers in which one is very big and the other very small, the approximation error becomes higher. If I do not remind wrong, this is what is called ill-conditioned problem in numerical analysis. That is why is important to normalise the features we have before send them to the classifier. If we do not do it, in this situation, the ZCR dominates on the AP.

```
% Voiced/unvoiced binary classification
clc, clear all, close all hidden

%% 1) Load the file 'voiced.wav'
[x, Fs] = audioread('voiced.wav');

%% 2) Define all windowing parameters for window extraction
%% length = 40 ms
%% spacing = 10 ms
% parameters in samples
length_s = round(40e-3*Fs);
hop_size_s = round(10e-3*Fs);
window = hamming(length_s);

%% 3) Extract ZCR and audio power for each window of the voiced sound
% number of windows
N = floor(length(x)/hop_size_s) - 4;

% container for features
feature = zeros(2, N); %row 1 zcr and row 2 the othe parameter

% loop over windows of x_v
for n=1:N

    % extract one audio window
    start_index = (n - 1) * hop_size_s + 1;
    stop_index = start_index + length_s - 1;
    frame = x(start_index : stop_index) .* window; %point wise product
    % compute features
    %ZCR = zerocrossrate(frame);
    ZCR = Fs/(2*length_s)*sum(abs(diff(sign(frame))));%the element of a vector and the next one.
    AP = 1/length_s * sum((abs(frame)).^2);

    feature(:, n) = [AP; ZCR];
end

%% 4) Plot in time on three subplots: the waveform, the ZCR and the AP

time = 0 : 1/Fs : (length(x) - 1)/Fs;
time_feature = (0:N-1)*hop_size_s/Fs;
figure;
subplot(3, 1, 1);
plot(time, x);
subplot(3, 1, 2);
plot(time_feature, feature(1,:));
subplot(3, 1, 3);
plot(time_feature, feature(2,:));
```

We have to check if it is mono or stereo signal. We can do it just looking at the size of x in the workspace. We notice that x has just one column, so it is a mono signal.

Since length\_s and audio\_size\_s are samples, it is better to round them to be sure to obtain an integer value instead of a decimal one.

Another solution to find the number of windows.

To make things clearer, we define the starting and the ending points.

We implement the zero crossing rate function without using the predefined one.

The semicolon instead of the comma since they are column.

We need a different time axis for the features since we have a different sampling period. We define it in a smarter way.

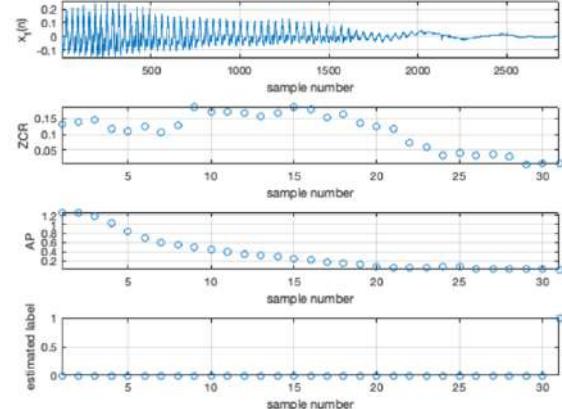
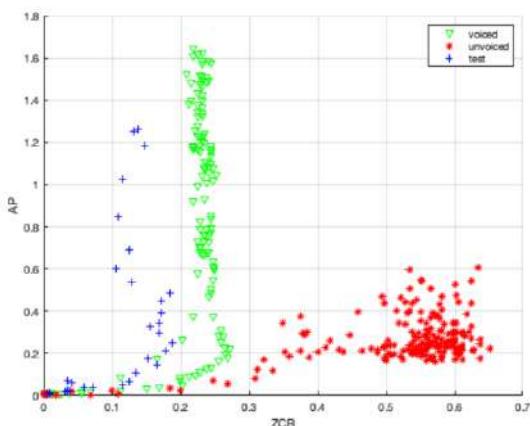
```
figure;
plot(feature(1, :), feature(2, :), 'd');
title("Feature space");
xlabel("AP");
ylabel("ZCR");
```

The last argument tells MATLAB to plot only the points.

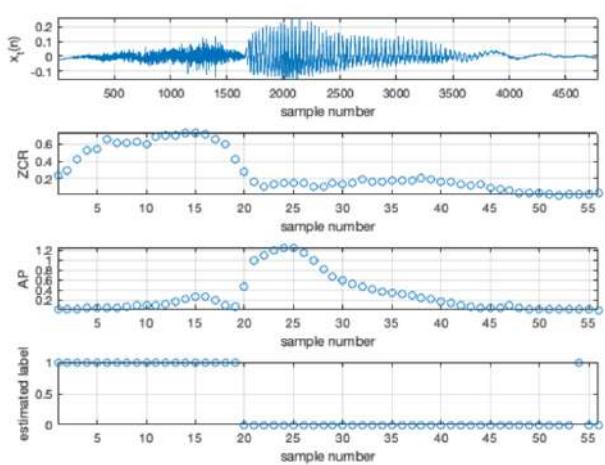
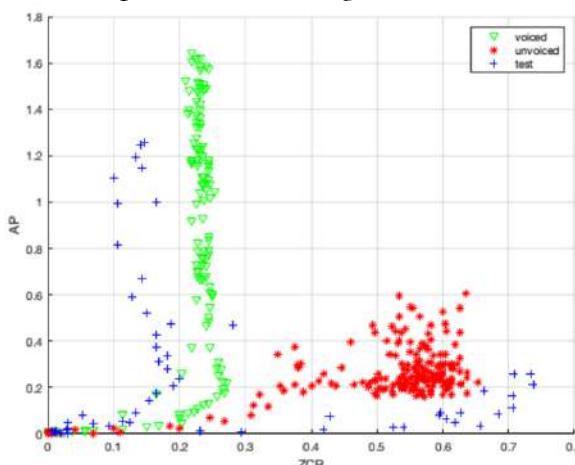
## Exercise 2 - KNN Classification

Deploy / test the system:

- Using the same parameters of exercise 1, extract ZCR and AP features from the test signal
- Plot test feature vectors in the feature space together with training features
- Using KNN with  $K=3$ , classify each window of the test signal separately
- Plot the test waveform, the ZCR for each window, the AP for each window, and the estimated label for each window



- Is the test sound a voiced or unvoiced one? Listen to it and check
- Repeat with 'test\_long.wav'



We do the same for the unvoiced signal. At the end we consider the unknown file.

```
% Voiced/unvoiced binary classification
clc, clear all, close all hidden

%% 1) Load the files 'voiced.wav', 'unvoiced.wav' and 'test'.wav
[x_v, Fs] = audioread('voiced.wav');
[x_u] = audioread('unvoiced.wav');
[x_t] = audioread('test.wav');
%[x_t] = audioread('test_long.wav');
```

```

%% 2) Define all windowing parameters for window extraction
%%     length = 40 ms
%%     spacing = 10 ms
% parameters in samples
frame_length = floor(40e-3 * Fs);
frame_spacing = floor(10e-3 * Fs);

% generate Hamming window
win = hamming(frame_length);

%% 3) Extract ZCR and audio power for each window of the training voiced sound
% number of windows
N = floor((length(x_v) - frame_length)/frame_spacing) + 1; %number of frames

% container for fetures
f_v = zeros(N, 2);

% loop over windows of x_v
for n=1:N

    % extract one audio window
    frame = x_v((n-1)*frame_spacing+1 : (n-1)*frame_spacing + frame_length);
%select the frame
    frame = frame .* win; %apply Hamming window

    % compute features
    f_v(n,1) = sum(abs(diff(frame>0))) / frame_length; %another way to
implement the zero crossing rate
    f_v(n,2) = sum(frame.^2);

end

%% 4) Extract ZCR and audio power for each window of the training unvoiced
sound
% number of windows
N = floor((length(x_u) - frame_length)/frame_spacing) + 1; %number of frames

% container for fetures
f_u = zeros(N, 2);

% loop over windows of x_v
for n=1:N

    % extract one audio window
    frame = x_u((n-1)*frame_spacing+1 : (n-1)*frame_spacing+frame_length);
%select the frame
    frame = frame .* win; %apply Hamming window

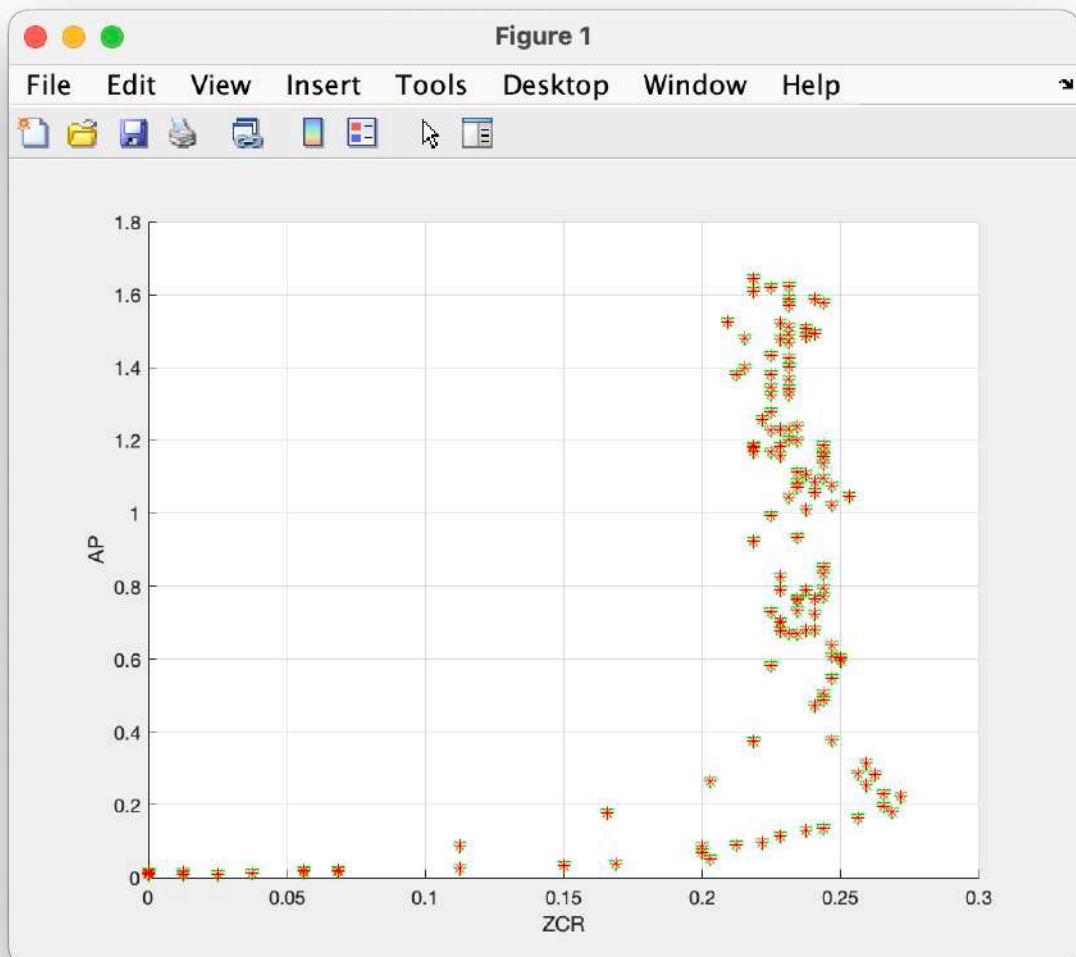
    % compute features
    f_u(n,1) = sum(abs(diff(frame>0)))/frame_length;
    f_u(n,2) = sum(frame.^2);

end

%% 5) Plot training features in the feature space using two different colors
figure(1), hold on
plot(f_v(:, 1), f_v(:, 2), 'vg')

```

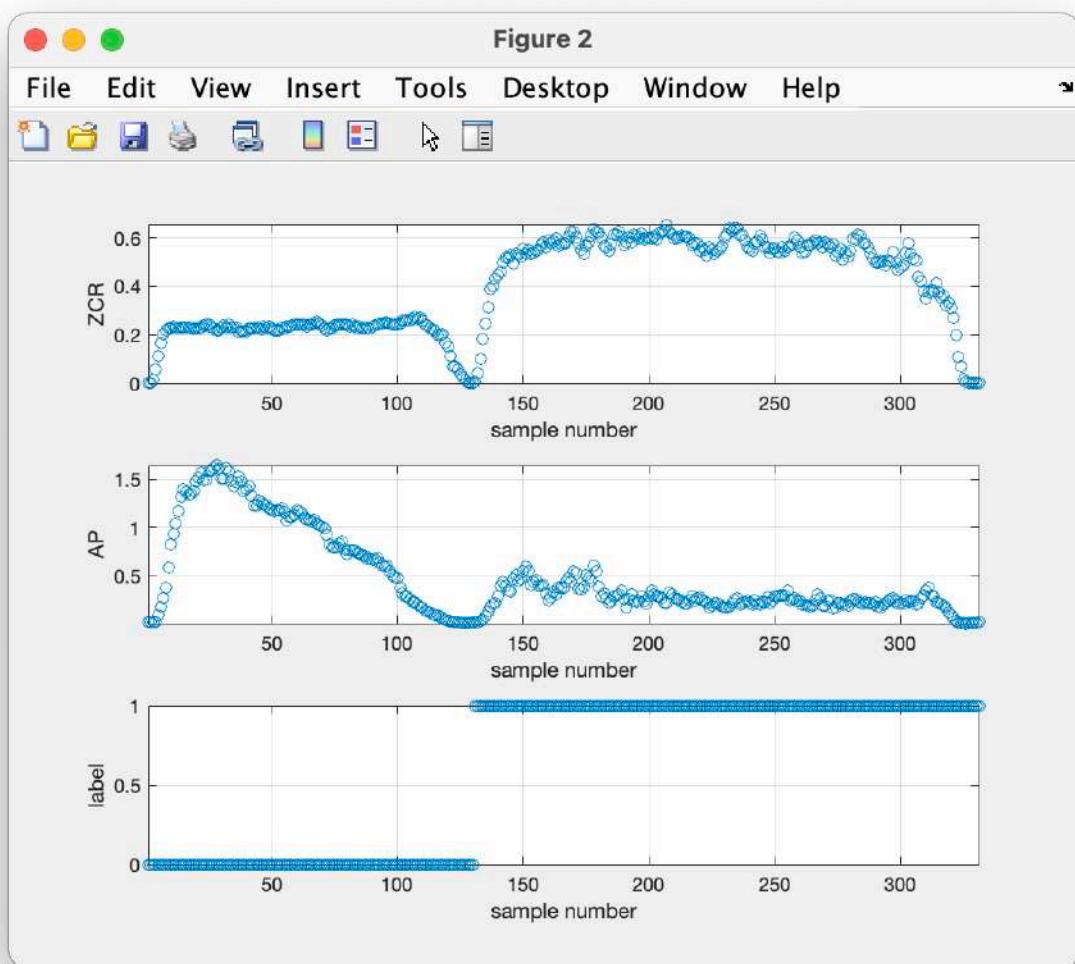
```
plot(f_v(:, 1), f_v(:, 2), '*r')
xlabel('ZCR')
ylabel('AP')
grid('on')
```



```
%% 6) Build training (feature, label) database for KNN classification
% concatenate features
f_db = [f_v; f_u];

% concatenate labels
y_db = [zeros(length(f_v), 1); ones(length(f_u), 1)];

% plot in three graphs: ZCR, AP, labels
figure(2)
subplot(3,1,1), plot(1:length(f_db(:, 1)), f_db(:, 1), 'o'), grid on,
xlabel('sample number'), ylabel('ZCR'), axis tight
subplot(3,1,2), plot(f_db(:, 2), 'o'), grid on, xlabel('sample number'),
ylabel('AP'), axis tight
subplot(3,1,3), plot(y_db, 'o'), grid on, xlabel('sample number'),
ylabel('label'), axis tight
```



```

%% 7) Extract ZCR and audio power for each window of the test sound
% number of windows
N = floor((length(x_t) - frame_length)/frame_spacing) + 1; %number of frames

% container for fetures
f_t = zeros(N, 2);

% loop over windows of x_v
for n=1:N

    % extract one audio window
    frame = x_t((n-1)*frame_spacing + 1 : (n - 1)*frame_spacing+frame_length);
%select the frame
    frame = frame .* win; %apply hamming window

    % compute features
    f_t(n, 1) = sum(abs(diff(frame > 0)))/frame_length;
    f_t(n, 2) = sum(frame .^ 2);
end

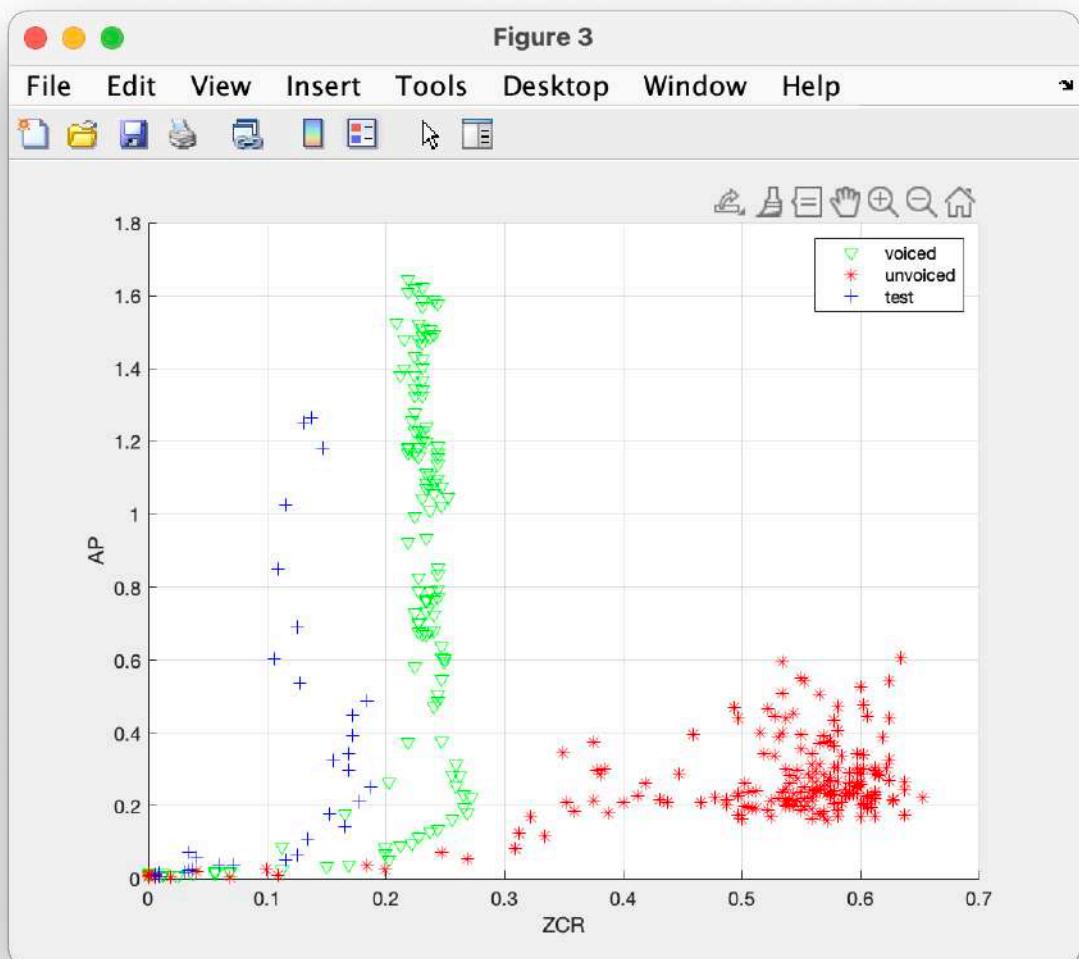
%% 8) Add test features to the feature space plot
figure(3), hold on

```

```

plot(f_v(:, 1), f_v(:, 2), 'vg')
plot(f_u(:, 1), f_u(:, 2), '*r')
plot(f_t(:, 1), f_t(:, 2), '+b')
legend('voiced', 'unvoiced', 'test')
xlabel('ZCR')
ylabel('AP')
grid('on')

```



```

%% 9) Classify each window of the test audio using KNN (K = 3)
K = 3;

% initialize container for estimated labels
y_hat = zeros(length(f_t), 1);

% loop over all windows
for f_idx = 1:length(f_t)
    % select feature vector for this window
    f = f_t(f_idx, :);

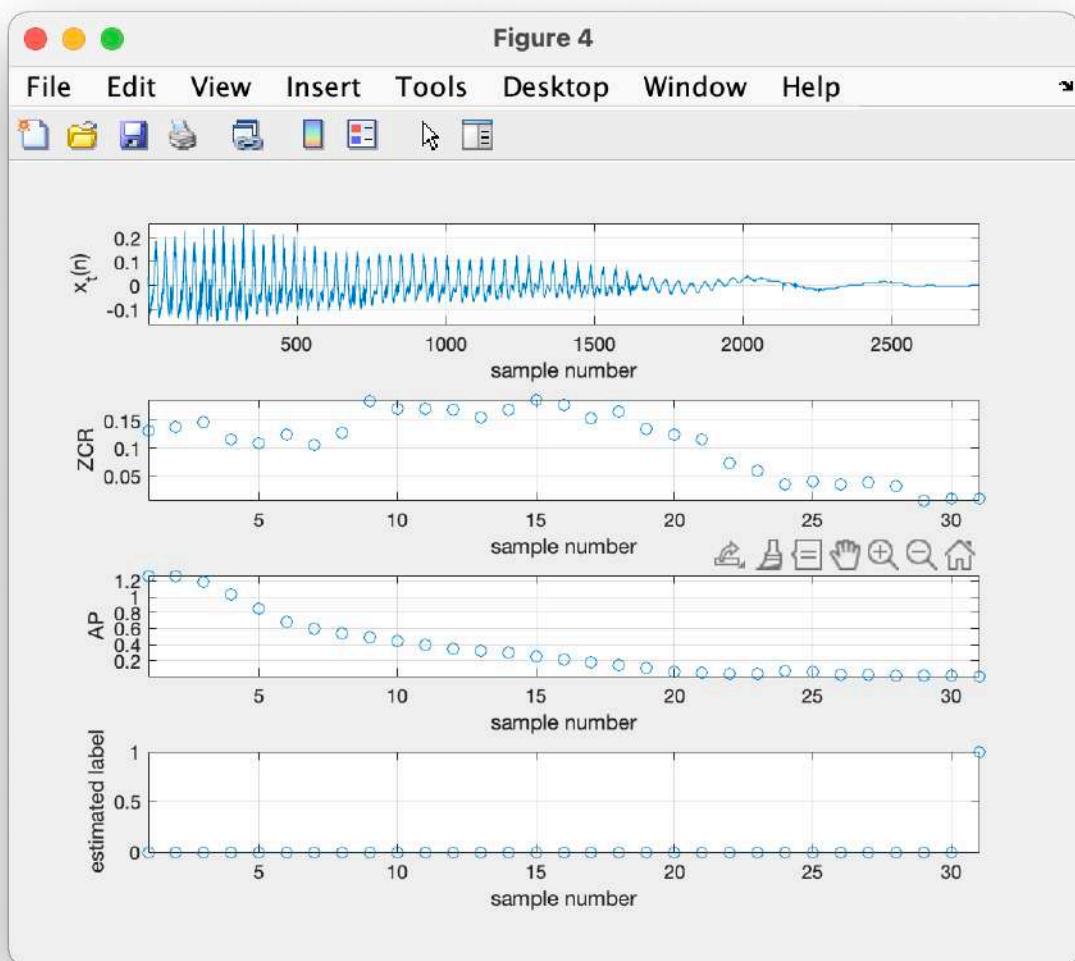
    % compute distance between test feature and all training features
    distance = sqrt(sum((f_db - f).^2, 2));

```

```
% sort distances and select classes from K closest features
[val, idx] = sort(distance);
classes = y_db(idx(1:3));

% apply majority voting to estimate the class of the test window
y_hat(f_idx) = mode(classes);
end

%% 10) Plot test waveform, features, and estimated class in time
figure(4)
subplot(411), plot(x_t), grid on, xlabel('sample number'), ylabel('x_t(n)'), axis tight
subplot(412), plot(f_t(:, 1), 'o'), grid on, xlabel('sample number'), ylabel('ZCR'), axis tight
subplot(413), plot(f_t(:, 2), 'o'), grid on, xlabel('sample number'), ylabel('AP'), axis tight
subplot(414), plot(y_hat, 'o'), grid on, xlabel('sample number'), ylabel('estimated label'), axis tight
```



## Microphone arrays - Spatial methods for DOA estimation

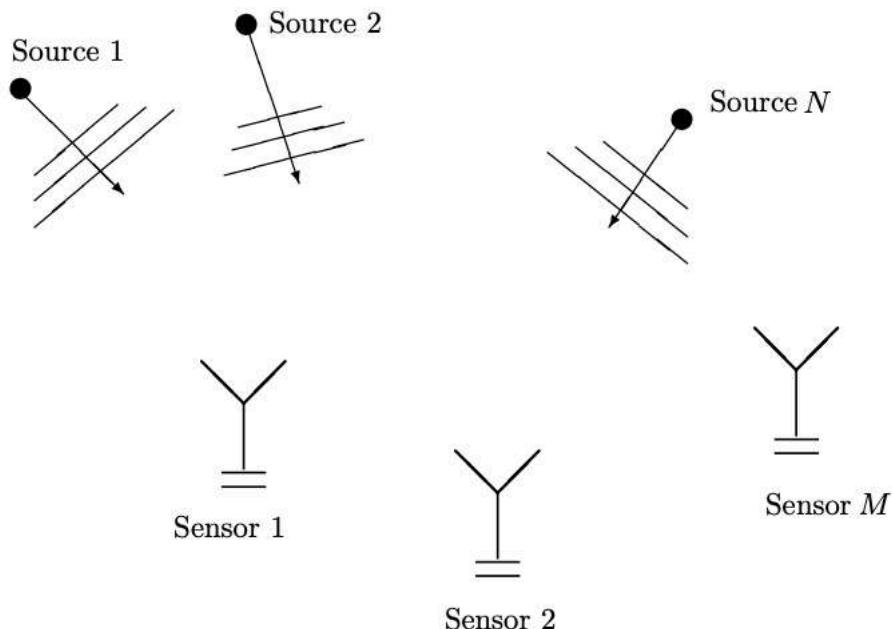
In today's lecture we will see how to exploit the properties of the Fourier transform to estimate the position of a source. There are several techniques to do it, today we will focus on two of the narrow-band approaches:

- Spatial filtering (non-parametric approaches)
- Parametric model

### Setting

First of all, we start defining the scenario:

we want to detect and locate  $N$  radiating sources by using an array of  $M$  passive microphones. In order to solve the problem, we try to understand how the acoustic energy is distributed over the space. By comparing the information on spatial energy distribution coming from different sensors, we are able to locate the sources.



In order to proceed, we have to make some assumptions

- Sources are located in the far-field. From which derives that
  - wavefronts are planar
  - only Directions Of Arrivals (DOAs) characterise the sources
 This assumption is essential: the acoustic waves propagate spherically (in every direction) and if we are not far enough we are not able to identify which is the precise direction from which the sound comes.
- Bidimensional geometry: sources and microphones lie on the same plane
 

This assumption is made just to simplify the derivations.
- No reverberation: we assume to be in the so-called free-field, without walls or surfaces on which the sound can bump. Despite being this one an assumption, it is true that, usually, the strongest component in the sound field is the direct one. However there are specific situations in which we cannot neglect the reverberation effects.

- No dispersion: the propagation is assumed just as a delay, without any modification in amplitude. This means that the amplitude of the wave in  $x_1$  at the time instant  $t_1$  will be the same that it will have at time instant  $t_2 = t_1 + \tau$  in the position  $x = x_1 + c\tau$  where  $c$  is the propagation velocity of the sound wave.
- Homogeneous propagation: the state variables of the system remain somehow constant (if we neglect the perturbation generated by the motion of the wave).

## Data Model

We assume, for the moment, the presence of a single source, impinging from a certain Direction Of Arrival (DOA) denoted as  $\theta$ . Our goal is to identify that direction of arrival.

The information related to the source can be retrieved from the sensors around it. The output of the  $k$ th sensor can be modelled as

$$\bar{y}_k(n) = h_k(n) * \bar{s}(n - \tau_k) + \bar{e}_k(n)$$

Where:

- $\bar{s}(n)$  is the source signal, as measured at a reference point
- $h_k(n)$  is the impulse response of the sensor
- $\tau_k$  is the propagation delay between the reference point and the sensor
- $\bar{e}_k(n)$  is the additive noise at the sensor

As we said before, the propagation in the signal expression appears just as a delay term  $\tau_k$ . Therefore, given a reference point and a polar coordinates system, the information about the position  $\theta$  is contained into the delays of the different sensors. This means that in order to find  $\theta$  we have to estimate these delays.

Simply speaking, if we have a wave that comes from right and two sensors (one on the right and one on the left), the signal will come before to the sensor on the right and then to the sensor on the left since the former is closer to the source. Given the fact that there will be a certain lag time between the moment in which the signal will be perceived from the right sensor and the moment in which it will be perceived from the left one, we can deduce that the signal comes from the right.

As we said, we will see methods for narrow band signal. This means that we consider just one carrier frequency  $\omega_c$ . It is a simplified model from which we can derive more complex models for signals that have a frequency that changes in time or for those ones that have more than one frequency since they are nothing but the result of a sum of simplified models. This is the basic idea behind the Fourier Transform. Thank to linearity, we can derive each signal from a sum of sinusoid. Therefore, if we are able to conduct our analysis for a sinusoidal signal, we can analyse every type of signal. This is why we will focus on just one frequency. By looking at the system in matrixial way, we are considering just one row of the STFT.

$$\bar{s}(n) = \alpha(n) \cos[\omega_c n + \phi(n)]$$

We extract  $L$ -long frames using a rectangular window and unitary hop-size:

$$s_t(n) = s(n)w(n - t) \quad w(n) = \begin{cases} 1, & 0 \leq n \leq L - 1 \\ 0, & \text{otherwise} \end{cases}$$

For each frame, we compute the DTFT

$$S_t(\omega) = DTFT\{s_t(n)\} = \sum_{n=-\infty}^{\infty} s_t(n)e^{-j\omega n} = ?? \underset{T=1}{STFT}\{s(n)\} = \sum_n \underbrace{w(n)}_{=1} s(t_i + n)e^{-j\omega n}$$

$$S_t(\omega) = \sum_{n=-\infty}^{\infty} s_t(n)e^{-j\omega n} \quad \text{Note that this corresponds to the STFT of } s(n), \\ \text{computed with unitary time resolution}$$

As we said before, we work at the source frequency, so we pose  $\omega = \omega_c$  obtaining

$$\begin{aligned} S_t(\omega = \omega_c) &= \sum_{n=-\infty}^{\infty} s_t(n)e^{-j\omega_c n} \\ &\stackrel{s_t(n)=s(n)w(n-t)}{=} \sum_{n=-\infty}^{\infty} s(n)w(n-t)e^{-j\omega_c n} = \\ &\stackrel{w(n)=\begin{cases} 1, & 0 \leq n \leq L-1 \\ 0, & \text{otherwise} \end{cases}}{=} \sum_{n=t}^{t+L-1} \underbrace{\alpha(n)}_{\substack{\text{Amplitude} \\ \text{Modulation}}} \cos \left[ \omega_c n + \underbrace{\phi(n)}_{\substack{\text{Phase} \\ \text{Modulation}}} \right] e^{-j\omega_c n} = \\ &\stackrel{w(n-t)=\begin{cases} 1, & 0 \leq n-t \leq L-1 \\ 0, & \text{otherwise} \end{cases}}{=} \\ &\stackrel{w(n-t)=\begin{cases} 1, & t \leq n \leq t+L-1 \\ 0, & \text{otherwise} \end{cases}}{=} \\ &\text{And } s(n)=\alpha(n) \cos[\omega_c n + \phi(n)] \end{aligned}$$

$$\begin{aligned} &n_{up}=l_{up}+t=t+L-1 \\ &\stackrel{\substack{\text{let } n=l+t \\ l=n-t}}{=} \sum_{l=t}^{l_{up}=L-1} \alpha(l+t) \cos[\omega_c \cdot (l+t) + \phi(l+t)] e^{-j\omega_c(l+t)} = \\ &= \sum_{l=0}^{l_{up}=L-1} \alpha(l+t) \cos[\omega_c(l+t) + \phi(l+t)] e^{-j\omega_c(l+t)} = \quad \boxed{\text{Demodulation}} \\ &= e^{j\omega_c t} \sum_{l=0}^{L-1} \alpha(l+t) \cos[\omega_c(l+t) + \phi(l+t)] e^{-j\omega_c l} \quad \approx \\ &\quad \text{We assume that the amplitude } \alpha(n) \\ &\quad \text{and phase } \phi(n) \text{ are slowly varying,} \\ &\quad (\text{almost constant over the} \\ &\quad \text{observation window}) \\ &\quad \alpha(t) \approx \alpha(t+l) \\ &\quad \phi(t) \approx \phi(t+l) \\ &\quad \text{for } l=0, 1, \dots, L-1 \\ &\quad \text{Simply speaking, they do not change} \\ &\quad \text{in the window, but they change} \\ &\quad \text{between different windows.} \\ &\approx e^{-j\omega_c t} \sum_{l=0}^{L-1} \alpha(t) \cos[\omega_c(l+t) + \phi(t)] e^{-j\omega_c l} = \\ &= e^{-j\omega_c t} \alpha(t) \sum_{l=0}^{L-1} \cos[\omega_c l + \omega_c t + \phi(t)] e^{-j\omega_c l} = \\ &= e^{-j\omega_c t} \alpha(t) \sum_{l=0}^{L-1} \cos[\omega_c l + \omega_c t + \phi(t)] e^{-j\omega_c l} \quad \approx \quad e^{-j\omega_c t} \alpha(t) e^{j[\omega_c l + \omega_c t + \phi(t)]} e^{-j\omega_c l} = \\ &\quad \text{we consider the STFT} \\ &= e^{-j\omega_c t} \alpha(t) e^{j[\omega_c t + \phi(t)]} = \alpha(t) e^{j\phi(t)} \end{aligned}$$

As the signal is narrowband and centred at the known frequency  $\omega_c$ , the previous expression can be interpreted as function of the time variable only.

Anyway, at the end, this series of passages will simplify our notation. We can define the signal  $s(t)$  (time dependency) as the amplitude and phase modulation of the signal at a specific carrier frequency

$$s(t) \triangleq S_t(\omega_c) = \alpha(t)e^{j\phi(t)}$$

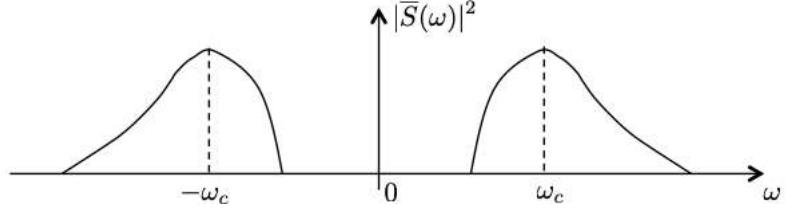
in other words, we are implicitly working in the STFT domain at a single frequency value corresponding to  $\omega = \omega_c$

In order to have a geometrical interpretation on what we are doing, we can notice that the computation of the STFT can be interpreted also as a demodulation operation, followed by a low-pass filtering of the signal.

1. We start from the signal  $\bar{s}(n) = \alpha(n) \cos[\omega_c n + \phi(n)]$  and we apply Euler's formula:

$$\begin{aligned} \bar{s}(n) &= \alpha(n) \cos[\omega_c n + \phi(n)] \stackrel{\text{Euler}}{=} \alpha(n) \frac{e^{j[\omega_c n + \phi(n)]} + e^{-j[\omega_c n + \phi(n)]}}{2} = \\ &= \frac{1}{2} \alpha(n) e^{j[\omega_c n + \phi(n)]} + \frac{1}{2} \alpha(n) e^{-j[\omega_c n + \phi(n)]} \end{aligned}$$

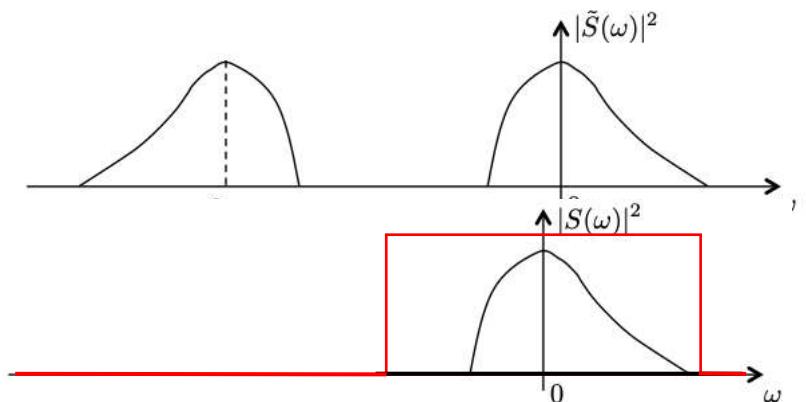
Reminding that a multiplication for a complex exponential in time domain corresponds to a shifting of the spectrum in frequency domain we will obtain what we can see in the figure on the right.



2. At this point we demodulate the signal. This means that we have to multiply to the signal a complex exponential that could shift back toward the origin the spectrum.

$$\begin{aligned} \tilde{s}(n) &= \bar{s}(n) \cdot 2e^{-j\omega_c n} = \frac{1}{2} \alpha(n) e^{j[\omega_c n + \phi(n)]} \cdot 2e^{-j\omega_c n} + \frac{1}{2} \alpha(n) e^{-j[\omega_c n + \phi(n)]} \cdot 2e^{-j\omega_c n} = \\ &= \alpha(n) e^{j[\omega_c n - \omega_c n + \phi(n)]} + \\ &\quad + \alpha(n) e^{-j[\omega_c n + \omega_c n + \phi(n)]} = \\ &= \alpha(n) e^{j\phi(n)} + \\ &\quad + \alpha(n) e^{-j[2\omega_c n + \phi(n)]} \end{aligned}$$

3. Now we low-pass-filter the demodulated signal to remove the spectral component at the higher frequency.



As we can see, we obtain a magnitude and phase derivation around the base band and not more around the carrier frequency.

We apply these derivations to our signal model:

$$y_k(n) = h_k(n) * s(n - \tau_k) + e_k(n) = H_k(\omega_c) * s(t)e^{-j\omega_c\tau_k} + e_k(t)$$

Where  $H_k(\omega_c)$  is the frequency response of the  $k$ -th microphone evaluated at  $\omega = \omega_c$ .

Since we have a multichannel acquisition, it is better to express the equation in vectorial form. In this way we are able to model the complete array system for a single source with a single vectorial equation:

$$\underline{y}(t) = \underline{a}(\theta)s(t) + \underline{e}(t)$$

Where:

The Array Vector is represented by      The Propagation Vector by      And, finally, the error Vector by

$$\underline{y}(t) = \begin{pmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_M(t) \end{pmatrix} \quad \underline{a}(\theta) = \begin{pmatrix} H_1(\omega_c)e^{-j\omega_c\tau_1} \\ H_2(\omega_c)e^{-j\omega_c\tau_2} \\ \vdots \\ H_M(\omega_c)e^{-j\omega_c\tau_M} \end{pmatrix} \quad \underline{e}(t) = \begin{pmatrix} e_1(t) \\ e_2(t) \\ \vdots \\ e_M(t) \end{pmatrix}$$

that describes the position of the source expressed as a delay. (The position information is into the delay  $\tau$ ).

This model can be also generalised to multiple sources. We just need a propagation vector for each of the sources we have in the signal. Hence, having  $s_1(t), s_2(t), \dots, s_N(t)$  source signals, we will have also  $\theta_1, \theta_2, \dots, \theta_N$  source DOAs respectively related to  $\underline{a}(\theta_1), \underline{a}(\theta_2), \dots, \underline{a}(\theta_N)$  propagation vectors.

Basing on the fact that we are in free-field and the mixture of the signals at the microphone is simply the instantaneous mixture of the sound at the sources (ideal linear acoustic propagation hypothesis), we can safely use the superposition principle to derive the array model for the case of multiple sources.

$$\underline{y}(t) = \underline{\underline{A}} \underline{s}(t) + \underline{e}(t)$$

Where:

$\underline{\underline{A}} = (\underline{a}(\theta_1), \underline{a}(\theta_2), \dots, \underline{a}(\theta_N))$  is the  $M \times N$  propagation matrix (it is a row array of column array)

$\underline{s}(t) = \begin{pmatrix} s_1(t) \\ s_2(t) \\ \vdots \\ s_N(t) \end{pmatrix}$  is the source vector.

Practically speaking, the different sources will arrive to the microphone with a different propagation vector, and we will capture at the microphone simply the sum of this source components.

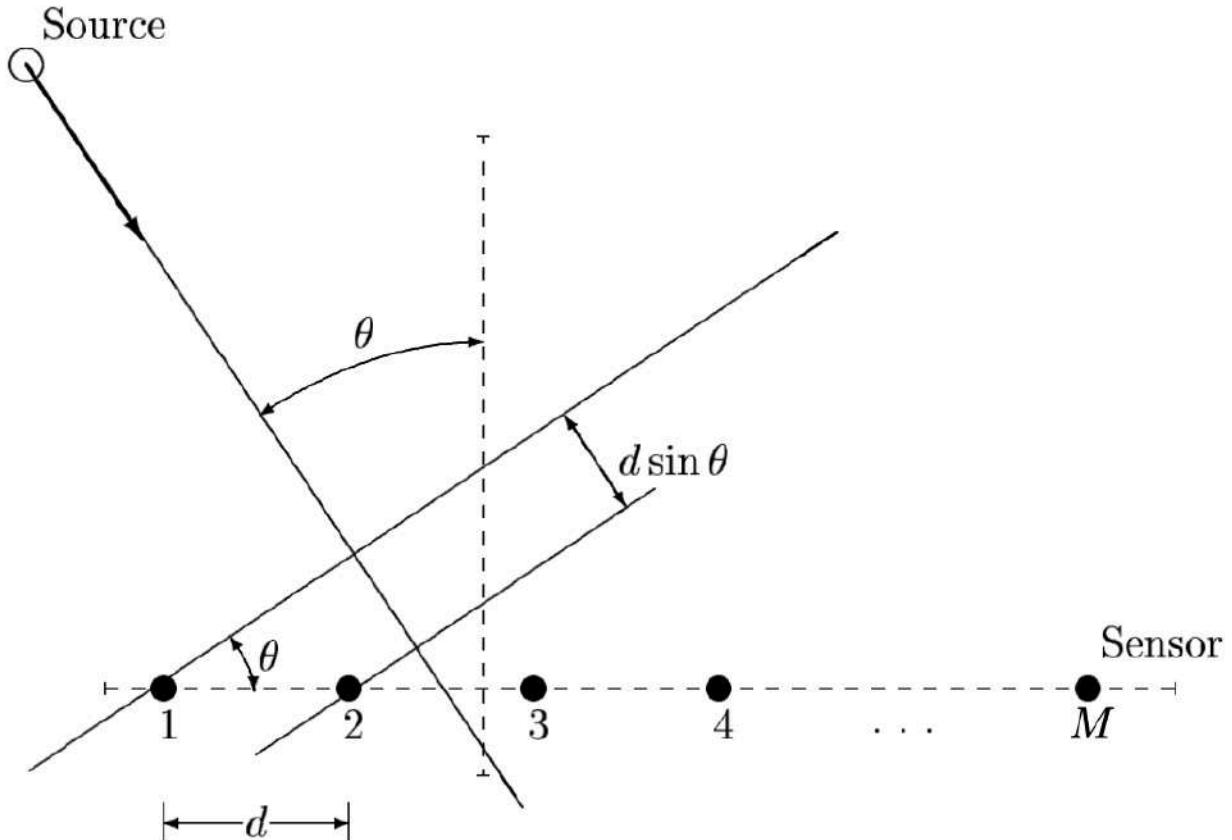
### Uniform linear array (ULA)

Up to now we have talked about the distribution of microphone in a very general manner. There are several shapes in which a microphone array can be assembled (planar, circular, spherical). It depends on the case: sometimes we need something that is easily designable and sometimes something that is more precise.

Anyway, for simplicity we will analyse the case of uniform linear array. This is nothing but an array of microphones in which the microphones are aligned and the distance between them is always the same.

In this particular situation there are some interesting properties that we can exploit:

<i>Align</i>	
<i>Line up</i>	Allineare



As we can easily notice from the image, we can find a relation between the angle  $\theta$  and the distance between the microphones  $d$ . In turns, the distance can be related to delay. In this way we can relate  $\theta$  and the delay term  $\tau$ . Exactly what we were looking for.

$$\tau_k = (k - 1) \frac{d \sin(\theta)}{c} \quad \text{for } \theta \in [-90^\circ, 90^\circ]$$

Where  $c \cong^1 340 \frac{m}{s}$  is the speed of sound in air.

We notice that we are using a relative approach: we consider a microphone (the first one) as the reference point, and we evaluate the delays with respect to this reference-microphone. In fact, in the delay expression, we have  $k - 1$  as for the first microphone ( $k = 1$ ) there is no delay with respect to itself.

All the microphones are assumed to be identical and omnidirectional, therefore we can assume:

$$H_1(\omega_c) = H_2(\omega_c) = \dots = H_M(\omega_c) = 1$$

Using the above expression, we can simplify that of the propagation vector:

$$\underline{a}(\theta) = \begin{pmatrix} H_1(\omega_c)e^{-j\omega_c\tau_1} \\ H_2(\omega_c)e^{-j\omega_c\tau_2} \\ \vdots \\ H_M(\omega_c)e^{-j\omega_c\tau_M} \end{pmatrix} \underset{\substack{\text{ULA} \\ \tau_k=(k-1)\frac{d \sin(\theta)}{c} \\ H_k(\omega_c)=1 \forall k \in [1, M]}}{=} \begin{pmatrix} 1 \cdot e^{-j\omega_c(1-1)\frac{d \sin(\theta)}{c}} \\ 1 \cdot e^{-j\omega_c(2-1)\frac{d \sin(\theta)}{c}} \\ \vdots \\ 1 \cdot e^{-j\omega_c(M-1)\frac{d \sin(\theta)}{c}} \end{pmatrix} = \begin{pmatrix} 1 \\ e^{-j\omega_c\frac{d \sin(\theta)}{c}} \\ \vdots \\ e^{-j(M-1)\omega_c\frac{d \sin(\theta)}{c}} \end{pmatrix}$$

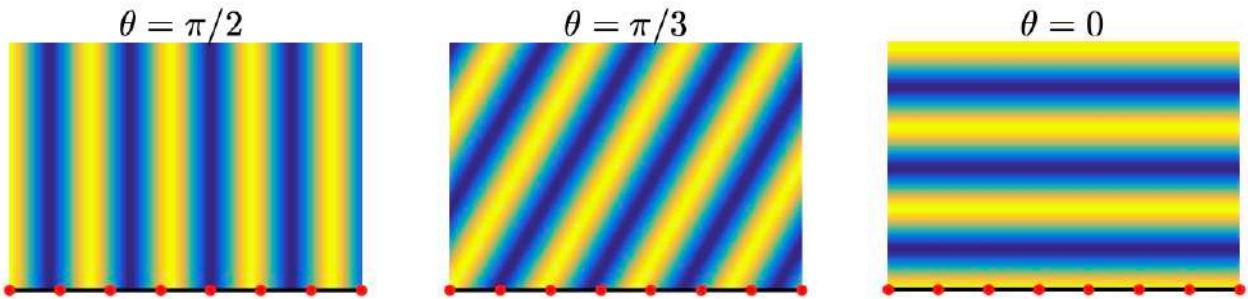
If we inspect the propagation vector, we can observe that basically contains the samples of a complex sinusoid. The word “sample” rings a bell in our head: sampling period and Nyquist condition. We will arrive to it in a second. For now, we just try to formalise the fact that we have to deal with complex sinusoid that depends on space.

$$e^{-jk\frac{\omega_c d \sin(\theta)}{c}} \quad \text{with } k = 0, 1, \dots, M-1$$

We define the frequency of the sinusoid as **spatial frequency**

$$\omega_s \triangleq \omega_c \frac{d \sin(\theta)}{c}$$

It is basically a frequency that tells us how the plane wave it is propagating in space. It is based on the temporal frequency (carrier frequency) and the direction of arrival  $\theta$ . In practice, we talk about sound field, that depends on time and varies in space. Placing our microphone arrays in space correspond to sample the wave field. The following pictures show graphically how the delay can change according to the  $\theta$  angle. The time variable is fixed at  $t = t_0$  as it was a picture.



In the three pictures, high pressure corresponds to yellow and blue to low pressure.

The red dots (the microphones) samples the wave in different points. The sampling condition is related to the distance between the microphones. According to Nyquist condition, in order to do a good sampling, we need to have at least two microphones for wavelength: and that's where the sampling frequency comes up. We can realise it just by looking at the pictures. If we have just one microphone for wavelength we bump into aliasing since we cannot be “precise enough” to follow the wave movements. Mathematically, this is called anti-aliasing condition.

Let us now formalise it.

We said that the complex sinusoid in the propagation vector  $\underline{a}(\theta)$  is sampled with a unitary sample period  $k = 0, 1, \dots, M-1$ .

It could be written, in fact, in function of multiples of the spatial frequency:

$$\underline{a}(\theta) = \begin{pmatrix} 1 \\ e^{-j\omega_c \frac{d \sin(\theta)}{c}} \\ \vdots \\ e^{-j(M-1)\omega_c \frac{d \sin(\theta)}{c}} \end{pmatrix}_{\omega_s \triangleq \omega_c \frac{d \sin(\theta)}{c}} = \begin{pmatrix} 1 \\ e^{-j\omega_s} \\ \vdots \\ e^{-j(M-1)\omega_s} \end{pmatrix}$$

Thus, for the sampling theorem, we must satisfy (spatial version of the Nyquist theorem):

$$|\omega_s| \leq \pi$$

This anti-aliasing condition impose that a certain distance has to be maintained between the microphones.

$$\begin{aligned} |\omega_s| &= \left| \omega_c \frac{d \sin(\theta)}{c} \right|_{\omega_s \triangleq \omega_c \frac{d \sin(\theta)}{c}} = \left| 2\pi \frac{\epsilon d \sin(\theta)}{\lambda} \right|_{\omega_c = 2\pi f_c = 2\pi \frac{c}{\lambda}} = \\ &= \left| 2\pi \frac{d \sin(\theta)}{\lambda} \right|_{\substack{2, \pi, \lambda, d > 0 \\ \text{are positive} \\ \text{by definition}}} 2\pi \frac{d}{\lambda} |\sin(\theta)| \leq \pi \rightarrow \\ &\rightarrow 2\pi \frac{d}{\lambda} |\sin(\theta)| \leq \pi \rightarrow d |\sin(\theta)| \leq \frac{\lambda}{2} \xrightarrow[\substack{0 \leq |\sin(\theta)| \leq 1 \\ \text{worst case} \\ |\sin(\theta)| = 1}]{} d \leq \frac{\lambda}{2} \end{aligned}$$

We conclude that the distance between two microphones has to be less than the half the wavelength.

## How to derive $\theta$

Up to now we have derived the model. Now we see some strategies to identify the DOA. We can exploit what we know about time-frequency models to create spatial-frequency models. As we saw in the introduction, the source DOA is directly related to the spatial frequency. Thus, DOA estimation can be brought to the problem of (spatial) frequency estimation since  $N$  sources impinging from different DOAs are characterised by different spatial frequencies. As we said before, there are two classes of methods that allows us to solve this problem:

- Non-parametric methods (a.k.a. spatial filtering):
  - DOA estimation without any assumptions about the covariance structure of the array data.
- Parametric methods:
  - DOA estimation based on a parametrisation of the covariance structure of the array data.

## Non-parametric method (Spatial Filtering)

The basic idea of spatial frequency is pretty simple. When we do filtering in time domain, what we get is a linear combination of the samples of our signal combines with the filter coefficients (where the filter coefficients are the scalar constants). The same thing is done here. We combine the microphone signal with some filter coefficients in order to obtain a desired response. Which is this desired response? The spatial frequency we are interested in (the one that correspond to the DOA)! Therefore, we have to design a filter  $h$  (a band pass filter) that will let pass the sound content related to that specific spatial frequency and attenuate

all the others. In this way we will enhance the signal coming from the direction  $\theta$  related to the frequency in which the band-pass filter is centred and attenuate the others. Simply explanation: it is nothing but the formalised version of the looking-listenscope of Rocket's equipment (for those ones who do not know the Little Einsteins, goes to 27:00 in the following link <https://www.youtube.com/watch?v=W1orEEY4uz8>).

Let us proceed in this way! Let us use the weights  $\underline{h} = [h_1, h_2, \dots, h_M]^H$  to obtain the spatial filtered signal  $y_F(t)$  as the result of the convolution between the filter and the signal  $y(t)$ . M is the number of microphones.

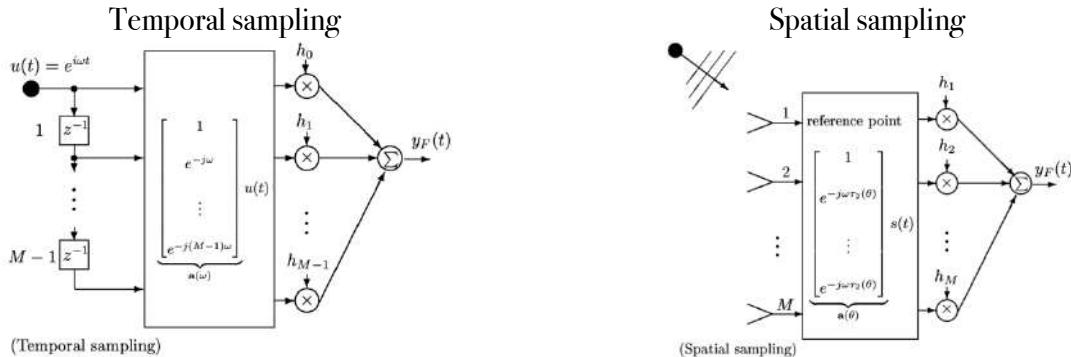
$$y_F(t) = \sum_{k=1}^M h_k \cdot y_k(t) \underset{\substack{\text{vectorial} \\ \text{form}}}{=} \underline{h}^H \underline{y}(t)$$

Let us assume that the filter  $h$  is band-pass, centred at  $\theta$  in the spatial domain. As we said, the power of the filtered signal should give a good indication of the energy coming from the direction  $\theta$ . What we imagine is that when we point the source directly on the microphone, the power will be maximised.

To know which is the value of  $\theta$  that maximise the energy (or rather, the value of  $\theta$  from which the sound comes so, where the sound is louder) we have previously to define the power of the filtered signal that is:

$$\begin{aligned} E\{|y_F(t)|^2\} &\underset{y_F(t)=\underline{h}^H \underline{y}(t)}{=} E\left\{\left|\underline{h}^H \underline{y}(t)\right|^2\right\} = E\left\{\underline{h}^H \underline{y}(t) \underline{y}^H(t) \underline{h}\right\} \underset{h \text{ is deterministic}}{=} \underline{h}^H E\left\{\underline{y}(t) \underline{y}^H(t)\right\} \underline{h} = \\ &\underset{\substack{R=E\{\underline{y}(t) \underline{y}^H(t)\} \\ \text{Covariance matrix} \\ \text{of the array data}}}{=} \underline{h}^H \underline{R} \underline{h} \end{aligned}$$

In order to better understand the system, let us see which are the analogies with the temporal filtering.



In this case we have a signal that is delayed and to which different coefficients are applied.

In this case we do not delay the samples since they are already “delayed” by the propagation vector.

Apart from the principle, it is basically the same and that is evident just by looking at the two formulas.

$\text{Temporal sampling}$ $y_F(t) = \sum_{k=0}^{M-1} h_k \cdot u(t - k) = \underbrace{[\underline{h}^H \underline{a}(\omega)]}_{\text{frequency response}} u(t)$	$\text{Spatial sampling}$ $y_F(t) = \sum_{k=1}^M h_k \cdot y_k(t) = \underbrace{[\underline{h}^H \underline{a}(\theta)]}_{\text{spatial response}} s(t)$
---	--

### *Delay-and-sum beamformer*

Therefore, at the end we have to design our filter  $\underline{h}^H(\bar{\theta})$  as an optimisation problem with a constrain that is: the sound coming from the direction  $\bar{\theta}$  has to pass undistorted, while the sound coming from other directions has to be attenuated. This is just the corresponding of the band pass filter in the spatial domain. The condition “let the signal coming from  $\bar{\theta}$  untouched” is formalised by the expression:

$$\underline{h}^H(\bar{\theta})\underline{a}(\bar{\theta}) = 1$$

While the condition “attenuate everything that does not come from  $\bar{\theta}$ ” is formalised in different ways depending on the model. Four a first approach, where we do not have information about how the sound is spatially distributed, we precede as it follows.

We assume that the signal is spatially white that means that the signal is coming from all the directions equally. It is just a parallelism: as in frequency domain the power spectrum of a white signal is flat, in spatial domain the power is equally distributed from every direction (the sound field comes from every direction with the same energy). Basing on this parallelism, we can say that the correlation matrix is equal to zero for every revenue except for the diagonal because we have that the microphone is correlated only with itself and not with the others. We remind that this is an assumption, in reality there is always a correlation between a same signal acquired in the same space and in the same moment by different microphones. We use this assumption simply for the nice result we obtain: the covariance matrix is equal to the identity matrix.

$$\underline{R} = \underline{I}_M$$

Since we want to attenuate the signals that do not come from  $\bar{\theta}$ , we have to minimise the power from all the directions with the constrain that  $\underline{h}^H(\bar{\theta})\underline{a}(\bar{\theta}) = 1$ . Mathematically speaking

$$\underline{h}(\bar{\theta}) = \underset{\underline{h}}{\operatorname{argmin}} \underline{h}^H \underline{h} \quad \text{subject to } \underline{h}^H(\bar{\theta})\underline{a}(\bar{\theta}) = 1$$

The problem has a close form solution that is given by

$$\underline{h}(\bar{\theta}) = \frac{\underline{a}(\bar{\theta})}{\underline{a}^H(\bar{\theta})\underline{a}(\bar{\theta})} \stackrel{\substack{\underline{a}^H(\bar{\theta})\underline{a}(\bar{\theta}) = |\underline{a}(\bar{\theta})| = M \\ \text{since } \underline{a} \text{ is the propagation vector}}}{=} \frac{\underline{a}(\bar{\theta})}{M}$$

Why  $\underline{a}^H(\bar{\theta})\underline{a}(\bar{\theta}) = |\underline{a}(\bar{\theta})| = M$ ?

$$\begin{aligned} \underline{a}^H(\bar{\theta})\underline{a}(\bar{\theta}) &= (1 \quad e^{+j\omega_s} \quad \dots \quad e^{+j(M-1)\omega_s}) \begin{pmatrix} 1 \\ e^{-j\omega_s} \\ \vdots \\ e^{-j(M-1)\omega_s} \end{pmatrix} = \\ &= 1 + e^{+j\omega_s}e^{-j\omega_s} + \dots + e^{+j(M-1)\omega_s}e^{-j(M-1)\omega_s} = \underbrace{1 + 1 + \dots + 1}_{M \text{ elements}} = M \end{aligned}$$

Why do numerator and denominator look like that?

Let us imagine having a signal derived from the propagation vector  $\underline{a}(\bar{\theta})$ . When we filter it using the filter we have just designed we get

$$\begin{aligned} \underline{h}^H(\bar{\theta})\underline{a}(\bar{\theta}) &\stackrel{\substack{\underline{a}^H(\bar{\theta}) \\ \underline{h}(\bar{\theta}) = \frac{\underline{a}(\bar{\theta})}{M}}}{=} \frac{\underline{a}^H(\bar{\theta})}{M}\underline{a}(\bar{\theta}) = \frac{M}{M} = 1 \end{aligned}$$

While when we change the direction (and consequently the propagation vector), we get, since the propagation vector is basically a set of delays (complex exponential), a sum of the delayed samples of the microphone with a given delay determined by the propagation vector  $\bar{\theta}$  that will build disruptive summation minimising the energy. That is why the filter is called delay-and-sum beamformer: we delay the signal using the propagation vector  $\underline{a}(\bar{\theta})$  and we sum the output of the delayed signal. As we just said, if we match the actual propagation vector the sum will be constructive and the energy maximised. On the other hand, disruptive.

If we plug this definition inside the power of the filtered output, we obtain the following formula:

$$E\{|y_F(t)|^2\} = \underline{h}^H(\bar{\theta}) \underline{R} \underline{h}(\bar{\theta}) \underset{\underline{h}(\bar{\theta}) = \frac{\underline{a}(\bar{\theta})}{M}}{\underset{\omega}{=}} \frac{\underline{a}^H(\bar{\theta})}{M} \underline{R} \frac{\underline{a}(\bar{\theta})}{M} = \frac{\underline{a}^H(\bar{\theta}) \underline{R} \underline{a}(\bar{\theta})}{M^2}$$

### Pseudo-spectrum

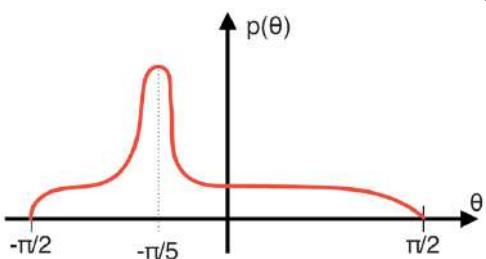
We conclude that in order to estimate the Direction of Arrival of the sources, we just have to evaluate the following function at all the spatial directions  $\theta$ :

$$p(\theta) = \frac{\underline{a}^H(\theta) \hat{\underline{R}} \underline{a}(\theta)}{M^2} \quad \text{with } \theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$$

Where:

- $\hat{\underline{R}} = \frac{1}{K} \sum_{t=1}^K \underline{y}(t) \underline{y}^H(t)$  is the sample estimate (we are not evaluating the expected value but an esteem made with a finite number of value) of the covariance matrix  $\underline{\underline{R}}$ ;
- The DOAs of the  $N$  sources are estimated as the  $N$  highest peaks of  $p(\theta)$

The function is called pseudo-spectrum since it recalls the frequency spectrum. It represents the energy content related to the direction of arrival (spatial frequencies).



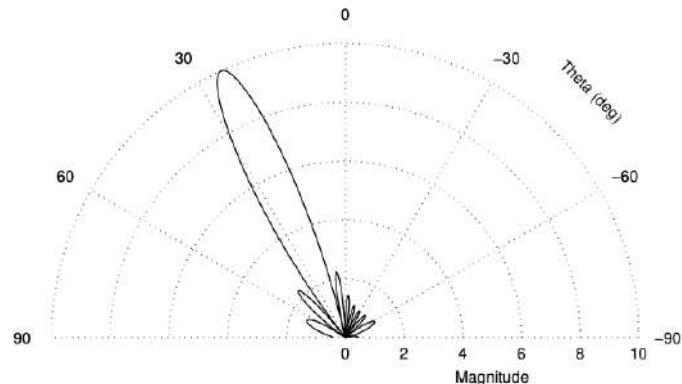
In the plot on the left, an example is reported. As we can notice there is a maximum in correspondence of  $\theta = -\frac{\pi}{5}$ . This happens because, with that propagation vector, the delay introduced by the filter generates constructive interference. This means that the source is placed in the radial direction  $\theta = -\frac{\pi}{5}$ .

As we can imagine, the number of peaks gives us an idea of the number of sources. If all the sources are placed along different  $\theta$ , the number of peaks correspond to the number of sources.

### Final observations

- The name delay-and-sum is due to the fact that the resulting filter weights in  $\underline{h}(\theta)$  correspond to pure delays.
- The effect of the filter is that of re-phasing the microphone signals accordingly to the propagation delays. The result is that we get:
  - constructive interference for the direction  $\bar{\theta}$
  - destructive interference for the other directions

- As the filter does not depend on the array data, the spatial response  $\underline{h}(\bar{\theta})\underline{a}(\theta)$  can be easily computed. Example, using an array with 10 microphones we get the following spatial frequency response for a filter designed with  $\bar{\theta} = 25^\circ$ . As we can notice, there is a peak in correspondence of  $\theta = 25^\circ$ . In this case the magnitude is equal to ten, the number of microphones  $M$ , since the axis it is not normalised ( $\underline{a}(\bar{\theta})$  is not divided by  $M$  in the definition of the filter). Furthermore, since the response of the filter for spatial frequencies that are not  $25^\circ$  is very low, the sound field that does not come from that  $25^\circ$  will be attenuated.



### *Resolution*

The narrowness of the lobe depends on the number of microphones you have. The narrower the lobe, the easier will be to identify the direction. Therefore, the more microphones you have, the higher will be the resolution of the system.

If we recall the analogies between frequency domain and spatial domain, we can link the spatial-domain concept of “the more microphone, the higher the resolution” to the frequency-domain concept “the more samples we have in time, the finer will be the frequency analysis we can do”. It is as if the microphones sampled the spatial domain.

Let us now formalise the concept. Given a uniform linear array whose microphones are spaced of  $d$  one with each other, we know that the overall aperture (the length of the array) is equal to  $l = (M - 1)d$ . Based on that, the width of each lobe is approximately given by  $\frac{\lambda}{l|\cos(\bar{\theta})|}$ . As we can see, the larger the aperture the narrower the width of the lobe. This time again we can find an analogy with what it happens in frequency domain: the larger the time window, the higher the resolution in frequency domain.

Anyway, we have to underline that actually there is also a connection and not just a formal analogy between spatial frequency and time frequencies. The spatial frequencies depend not only on  $\theta$  but also on  $\lambda$ , that in turns depends on  $\omega$ . This means that the resolution of the main lobe will depend both on the direction in which we are pointing the microphone and also on the frequency of the signal! That is why we have at the denominator  $\bar{\theta}$  and at the denominator  $\lambda$ . As we can notice, when  $\cos(\bar{\theta}) \rightarrow 0$  the function goes to  $\infty$ . This means that we are pointing towards the other points of the microphone arrays. While, when  $\cos(\bar{\theta}) = 1$  the microphone is directed the frontal direction. If we consider two sources with angular separation  $\Delta\theta = \theta_2 - \theta_1$  located around that frontal direction (so when  $\bar{\theta} = 0^\circ$  and  $\cos(\bar{\theta}) = 1$ ), we can affirm that they can be resolved by the function  $p(\theta)$  if:

$$\Delta\theta > \frac{\lambda}{l}$$

If this condition is not respected, we are not able to distinguish two sources. A single peak will be visible somewhere in the range  $[\theta_1, \theta_2]$  leading to inconsistent DOA estimation. Simply speaking, the sources are too close to understand their position without confusing them with each other. It is like when there are two people speaking and we are not close enough to understand who is speaking. We just hear a voice coming from that direction, but we cannot distinguish the two people.

## Capon Method

This was the basic spatial filtering model that is actually, by the way, very popular since the filter definition is very easy (the propagation vector in a given direction  $\underline{h}(\bar{\theta}) = \frac{\underline{a}(\bar{\theta})}{M}$ ). But this definition is as easy because we made a strong assumption: we neglected the covariance relation between microphones (we imposed that the covariance matrix is equal to the identity matrix  $\underline{R} = \underline{I}_M$ ). Actually, it is impossible that there is no correlation between the signal coming from microphones that are acquiring the same signal! Since we now consider the actual covariance matrix, the formulation of the optimisation problem becomes

$$\underline{h}(\bar{\theta}) = \operatorname{argmin}_{\underline{h}} \underline{h}^H \underline{R} \underline{h} \quad \text{subject to } \underline{h}^H(\bar{\theta}) \underline{a}(\bar{\theta}) = 1$$

The Capon method puts effort in attenuating the actual interferent (unwanted) sources, rather than attenuating the energy from all the possible locations.

Also this problem has a close form solution

$$\underline{h}(\bar{\theta}) = \frac{\underline{R}^{-1} \underline{a}(\bar{\theta})}{\underline{a}^H(\bar{\theta}) \underline{R}^{-1} \underline{a}(\bar{\theta})}$$

The definition is the same as before but with the spatial covariance matrix instead of the identity matrix. But hey! It is inverted. To precede, we have to assure that the matrix is not singular (so that is invertible).

At the end the localisation approach is the same as before. This higher resolution comes at the cost of some cons:

- The filter cannot be computed in advanced since it is signal dependent (the covariance matrix depends on the signal).
- We have to invert a matrix (operation that is computationally expansive).

At the end, the performance of Capone gives a better result (higher resolution) than the beamforming, but it costs too much. In fact, it is not so widely used since it is not worth the hassle.

---

	
<i><b>It is not worth the hassle</b></i>	Il gioco non vale la candela

---

## Parametric methods

There is also another class of techniques that lies on some observation we can do on the spatial covariance matrix of the signal instead of working with spatial frequencies. We try to estimate the DOAs with two different approaches. One based on the pseudo-spectrum and the other estimating directly  $\omega_s$ . In both cases we need to rely on some requirements:

- I. The number of sources  $N$  must be known in advance and it must be lower than the number of microphones. So

$$N < M$$

2. The source signals  $s_1(t), s_2(t), \dots, s_N(t)$  are such that their covariance matrix is non-singular (so that the signals are not so much correlated even though some correlation is admitted). So the rank of the covariance matrix of the signal must be equal to the number of sources

$$\text{rank}(\underline{\underline{R}}_s) = N \quad \text{where } \underline{\underline{R}}_s = E\{\underline{s}(t)\underline{s}^H(t)\}$$

Reminder: rank of a matrix

The rank of a matrix  $\underline{\underline{A}}$  is the dimension of the vector space generated (or spanned) by its columns.

This corresponds to the maximal number of linearly independent columns of  $\underline{\underline{A}}$ . This, in turn, is identical to the dimension of the vector space spanned by its rows. Rank is thus a measure of the "nondegenerateness" of the system of linear equations and linear transformation encoded by  $A$ . This means, in our case, that if the rank is equal to the number of sources, the number of linear independent rows (or columns) will be lower than  $N$ . If the rank is equal to the dimension all the matrix, all the rows and columns of that matrix will be linear independent with each others.

3. The sensor noise is spatially white, so the variance is always the same for all their components. We can formalise the concept by saying writing that

$$\underline{\underline{R}}_e = E\{\underline{e}(t)\underline{e}^H(t)\} = \sigma^2 \underline{\underline{I}}_M$$

4. Senor noise is uncorrelated to source signals  
 5. All the DOAs are different, thus they lead to different spatial frequencies

$$\omega_{s_1} \neq \omega_{s_2} \neq \dots \neq \omega_{s_N}$$

### Covariance model

With these assumptions we can observe the covariance matrix of the microphones can be written as follows:

$$\underline{\underline{R}} = \underbrace{\underline{\underline{A}} \underline{\underline{R}}_s \underline{\underline{A}}^H}_{\substack{\text{Sources} \\ \text{dependent}}} + \underbrace{\sigma^2 \underline{\underline{I}}_M}_{\substack{\text{Noise} \\ \text{dependent}}}$$

It is possible to decompose the covariance matrix in the sum of two components since the two components are uncorrelated. Also, the sources are uncorrelated (in fact, having imposed that the rank is equal to  $N$  and being the dimension of the matrix  $N$ , all its rows are linearly independent) so they can be decomposed in the multiplication between the propagation matrix  $\underline{\underline{A}} = [\underline{a}(\theta_1), \underline{a}(\theta_2), \dots, \underline{a}(\theta_N)]$  and other. When we evaluate the covariance matrix of the signal we end with  $E\{\underline{\underline{A}} \underline{s}(t)\underline{s}^H(t)\underline{\underline{A}}^H\} = \underline{\underline{A}} E\{\underline{s}(t)\underline{s}^H(t)\}\underline{\underline{A}}^H = \underline{\underline{A}} \underline{\underline{R}}_s \underline{\underline{A}}^H$ .

At this point, we can perform the eigenvalue decomposition of  $\underline{\underline{R}}$

Reminder: what is an eigenvalue?

Given a squared matrix  $\underline{\underline{A}}$ , we call  $\lambda \in \mathbb{C}$  an eigenvalue of  $\underline{\underline{A}}$  if it exists a vector  $\underline{v} \neq 0$  such that

$$\underline{\underline{A}} \underline{v} = \lambda \underline{v}$$

The vector  $\underline{v}$  is called eigenvector.

Practical approach: why do we perform eigenvalue decomposition? Because by knowing the eigenvalue we can obtain a matrix (by changing the bases) in which those eigenvalues are on a diagonal. We like to work with matrixes full of zeroes where numbers different from zero compare only on the diagonal.

What information they provide?

When applied to a square matrix, eigenvalues represent the values for which an eigenvector remains in the same direction after the matrix transformation. In other words, the eigenvector only gets scaled by a factor determined by the corresponding eigenvalue.

How can we perform eigenvalue decomposition (Eigendecomposition)?

It has to happen that  $\underline{\underline{A}} \underline{v} = \lambda \underline{v}$  has a solution with  $\underline{v} \neq 0$ . This means that  $\underline{\underline{A}} \underline{v} - \lambda \underline{v} = 0$  and that, in turns,  $(\underline{\underline{A}} - \lambda \underline{\underline{I}}) \underline{v} = 0$  has a solution for  $\underline{v} \neq 0$ . In order to obtain this result  $\underline{v} \in \ker(\underline{\underline{A}} - \lambda \underline{\underline{I}})$  and that is possible if and only if

$$\det(\underline{\underline{A}} - \lambda \underline{\underline{I}}) = 0$$

where  $\det(\underline{\underline{A}} - \lambda \underline{\underline{I}})$  is called characteristic polynomial.

*Example:*

$$\begin{aligned} \underline{\underline{A}} &= \begin{pmatrix} 4 & 2 \\ 1 & 3 \end{pmatrix} \rightarrow \underline{\underline{A}} - \lambda \underline{\underline{I}} = \begin{pmatrix} 4 & 2 \\ 1 & 3 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 4-\lambda & 2 \\ 1 & 3-\lambda \end{pmatrix} \\ \det(\underline{\underline{A}} - \lambda \underline{\underline{I}}) &= \det \begin{pmatrix} 4-\lambda & 2 \\ 1 & 3-\lambda \end{pmatrix} = (4-\lambda)(3-\lambda) - 2 = \lambda^2 - 7\lambda + 12 - 2 = \\ &= \lambda^2 - 7\lambda + 10 = (\lambda - 5)(\lambda - 2) = 0 \rightarrow \lambda = \begin{cases} 5 \\ 2 \end{cases} \end{aligned}$$

Once we have found the Eigenvalues and, consequently, the eigenvectors, we can factorise  $\underline{\underline{A}}$  as

$$\underline{\underline{A}} = \underline{\underline{Q}} \underline{\underline{\Lambda}} \underline{\underline{Q}}^{-1}$$

where  $\underline{\underline{Q}}$  is the square  $n \times n$  matrix whose  $i$ -th column is the eigenvector  $\underline{v}_i$  of  $\underline{\underline{A}}$ , and  $\underline{\underline{\Lambda}}$  is the diagonal matrix whose diagonal elements are the corresponding eigenvalues,  $\Lambda_{i,i} = \lambda_i$ .

Reminder: Canonical form and similar matrixes

We call **similar** two  $n$ -by- $n$  matrixes  $\underline{A}$  and  $\underline{B}$  if there exists an invertible  $n$ -by- $n$  matrix  $\underline{M}$  such that

$$\underline{B} = \underline{M}^{-1} \underline{A} \underline{M}$$

Similar matrices represent the same linear map under two (possibly) different bases, with  $\underline{M}$  being the change of basis matrix. What we do is take the matrix  $\underline{A}$  in a vectorial space in which it has a simpler representation and then we take it back to the original vectorial space.

When we look for the canonical form of a matrix, we simply seek a matrix  $B$  similar to  $A$  but as easy as possible at the same time. One of the possibilities to obtain it, it is by means of the diagonalisation: a process for which we find a matrix  $\underline{M}$  able to transform the matrix  $\underline{B}$  in a vectorial space in which it is diagonal  $\underline{\underline{A}}$ .

Properties of similar matrixes:

- They have the same characteristic polynomial
- They have the same eigenvalues
- They have the same determinant
- They have the same trace

Having performed the eigenvalue decomposition of the matrix  $\underline{R}$ , and having sorted the eigenvalues such that  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_M$ , we can rewrite the covariance matrix as:

$$\underline{R} = \underline{M} \underline{\Lambda} \underline{M}^H = (\underline{U} \underline{V}) \begin{pmatrix} \lambda_1 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & & & & \vdots \\ \vdots & & \lambda_N & & & \vdots \\ \vdots & & & \lambda_{N+1} & & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & \lambda_M \end{pmatrix} \begin{pmatrix} \underline{U}^H \\ \underline{V}^H \end{pmatrix}$$

Where:

- $\underline{U}$  is the matrix of the orthonormal eigenvectors associated to the first  $N$  (number of sources) eigenvalues  $\{\lambda_1, \dots, \lambda_N\}$
- $\underline{V}$  is the matrix of the orthonormal eigenvectors associated to the remaining eigenvalues that goes from  $N + 1$  to  $M$  (the number of microphones)  $\{\lambda_{N+1}, \dots, \lambda_M\}$
- $\underline{\Lambda}$  is the diagonal matrix of the eigenvalues.

Moreover, since the source signal and the noise are uncorrelated, we can rewrite the diagonal eigenvalue matrix making another decomposition:

$$\underline{\Lambda} = \begin{pmatrix} \lambda_1 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & & & & \vdots \\ \vdots & & \lambda_N & & & \vdots \\ \vdots & & & \lambda_{N+1} & & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & \lambda_M \end{pmatrix} = \underbrace{\underline{\Lambda}_s}_{\substack{\text{Eigenvalues related} \\ \text{to the source part}}} + \underbrace{\underline{\Lambda}_e}_{\substack{\text{Eigenvalues related} \\ \text{to the noise part}}} \quad \text{Since } \sigma^2 \underline{I}_M \text{ is already a diagonal matrix}$$

$\underline{A} \underline{R}_s \underline{A}^H \quad \sigma^2 \underline{I}_M \quad \underline{\Lambda}_e = \sigma^2 \underline{I}_M$

$$= \underline{\Lambda}_s + \sigma^2 \underline{I}_M$$

We also did the assumption that frequency come from different directions, so that the spatial frequencies are all different one with each other ( $\omega_{s_1} \neq \omega_{s_2} \neq \dots \neq \omega_{s_N}$ ). Due to this fact, also the propagation vectors will be different one with each other. This implies that the propagation matrix has a rank (maximum number of independent rows and columns) equal to  $N$  where  $N$  is the number of sources. Formally speaking, the propagation matrix is full rank

$$\text{rank}(\underline{\underline{A}}) = \text{rank}((\underline{a}(\theta_1), \underline{a}(\theta_2), \dots, \underline{a}(\theta_N))) = N$$

Moreover, since we supposed  $\text{rank}(\underline{\underline{R}_s}) = N$ , it must be that

$$\text{rank}(\underline{\underline{R}_s}) = \text{rank}(\underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H) = N$$

It follows the eigenvalues different from zero of  $\underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H$  are just the one from 1 to  $N$

$$\underline{\underline{\Lambda}}_s = \begin{pmatrix} \bar{\lambda}_1 & & & \\ & \ddots & & \\ & & \bar{\lambda}_N & \\ & & & 0 \\ & & & & \ddots \\ & & & & & 0 \end{pmatrix}$$

These eigenvalues are the one related to the source signal that is propagating through the microphone.

Why?

Time for another reminder!

The rank of matrix can be seen in three different ways (or rather, it represents three things that coincide):

- R-rank: maximum number of linear independent rows (dimension of the span generated by the rows)
- C-rank: maximum number of linear independent columns (dimension of the span generated by the columns)
- D-rank: maximum dimension of a minor (determinant of a smaller square matrix cut down from another bigger matrix) different from zero. In other words, maximum value  $r$  for which there exists a square submatrix with  $\text{Det} \neq 0$ .

In order to understand why if  $\text{rank}(\underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H) = N$ ,  $\underline{\underline{\Lambda}}_s$  is in that shape, we have to consider the D-rank vision of the rank. Whatever square matrix whose dimension is greater than  $N$  we take, the zero on the diagonal makes the determinant becomes zero. In fact

$$\underline{\underline{\Lambda}}_s = \begin{pmatrix} \bar{\lambda}_1 & & & \\ & \ddots & & \\ & & \bar{\lambda}_N & \\ & & & 0 \\ & & & & \ddots \\ & & & & & 0 \end{pmatrix}$$

the submatrix highlighted in red has dimension  $N \times N$ . This implies that the  $\text{rank}(\underline{\underline{\Lambda}}_s) = N$ .

We remind also that similar matrixes have same rank.

Let us demonstrate it.

Lemma (1): We will show the  $\text{rank}(\underline{\underline{A}} \underline{\underline{B}}) = \text{rank}(\underline{\underline{B}})$  if  $\underline{\underline{A}}$  is invertible.

Take some  $\underline{u} \in \ker(\underline{\underline{B}})$ . Then, since the kernel (also called the null-space) is by definition is the linear subspace of the domain which the map maps to the zero vector,  $\underline{\underline{B}} \underline{u} = 0$ ,

so  $\underline{\underline{A}} \underline{\underline{B}} \underline{u} = \underline{\underline{A}} (\underline{\underline{B}} \underline{u}) = \underline{\underline{A}} 0 = 0$ . Thus,  $\underline{u} \in \ker(\underline{\underline{A}} \underline{\underline{B}})$ , so  $\ker(\underline{\underline{B}}) \subseteq \ker(\underline{\underline{A}} \underline{\underline{B}})$ .

Conversely, for some  $\underline{u} \in \ker(\underline{\underline{A}} \underline{\underline{B}})$ ,  $(\underline{\underline{A}} \underline{\underline{B}}) \underline{u} = 0$ , so  $\underline{\underline{A}} (\underline{\underline{B}} \underline{u}) = 0$ . Thus, supposing that  $\underline{\underline{A}}$  is not singular so that is invertible,  $\underline{\underline{B}} \underline{u} = \underline{\underline{A}}^{-1} 0 = 0$ . This implies that  $\underline{u} \in \ker(\underline{\underline{B}})$ . Therefore,

$\ker(\underline{\underline{A}} \underline{\underline{B}}) = \ker(\underline{\underline{B}})$ , implies  $\text{null}(\underline{\underline{A}} \underline{\underline{B}}) = \text{null}(\underline{\underline{B}})$ , implies  $\text{rank}(\underline{\underline{A}} \underline{\underline{B}}) = \text{rank}(\underline{\underline{B}})$ , where the operator  $\text{null}()$  returns the element that are equal to zero of the argument.

Lemma (2): Show that:  $\text{rank}(\underline{\underline{A}} \underline{\underline{B}}) = \text{rank}(\underline{\underline{A}})$ , if  $\underline{\underline{B}}$  is invertible.

$$\text{rank}(\underline{\underline{A}} \underline{\underline{B}}) \stackrel{\substack{\equiv \\ \text{The result that } \text{rank}() \\ \text{returns can represent} \\ \text{both the R-rank and} \\ \text{the C-rank, so}}}{} \text{rank}((\underline{\underline{A}} \underline{\underline{B}})^T) = \text{rank}(\underline{\underline{B}}^T \underline{\underline{A}}^T) \stackrel{\substack{\equiv \\ \text{Lemma 1}}}{} \text{rank}(\underline{\underline{A}}^T) = \text{rank}(\underline{\underline{A}})$$

$\text{rank}(\underline{\underline{y}}) = \text{rank}(\underline{\underline{y}}^T)$

since  $\underline{\underline{B}}^T$  is non-singular.

Dem:

We have  $\text{rank}(\underline{\underline{X}} \underline{\underline{B}}) = \text{rank}(\underline{\underline{B}})$  from Lemma (1) and  $\text{rank}(\underline{\underline{A}} \underline{\underline{X}}) = \text{rank}(\underline{\underline{A}})$  from Lemma (2). If we consider  $\underline{\underline{X}}$  as the transformation matrix, we can write that  $\underline{\underline{X}}^{-1} \underline{\underline{A}} \underline{\underline{X}} = \underline{\underline{B}}$  implies  $\underline{\underline{A}} \underline{\underline{X}} = \underline{\underline{X}} \underline{\underline{B}}$ .

Therefore,

$$\text{rank}(\underline{\underline{A}} \underline{\underline{X}}) \stackrel{\substack{\equiv \\ \underline{\underline{A}} \underline{\underline{X}} = \underline{\underline{X}} \underline{\underline{B}}}}{} \text{rank}(\underline{\underline{X}} \underline{\underline{B}}) \xrightarrow{\substack{\text{rank}(\underline{\underline{A}} \underline{\underline{X}}) = \text{rank}(\underline{\underline{A}}) \\ \text{rank}(\underline{\underline{X}} \underline{\underline{B}}) = \text{rank}(\underline{\underline{B}})}} \text{rank}(\underline{\underline{A}}) = \text{rank}(\underline{\underline{B}})$$

q.e.d.

So, what we get is

$$\underline{\underline{\Lambda}} = \begin{pmatrix} \lambda_1 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & & & & \vdots \\ \vdots & & \lambda_N & & & \vdots \\ & & & \lambda_{N+1} & & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & \lambda_M \end{pmatrix} = \underline{\underline{\Lambda}}_s + \sigma^2 \underline{\underline{I}}_M =$$

$$\begin{aligned}
&= \begin{pmatrix} \bar{\lambda}_1 & & & \\ & \ddots & & \\ & & \bar{\lambda}_N & \\ & & & 0 \\ & & & & \ddots \\ & & & & & 0 \end{pmatrix} + \begin{pmatrix} \sigma^2 & & & & & \\ & \ddots & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & & & & \ddots & \\ & & & & & \sigma^2 \end{pmatrix} = \\
&= \begin{pmatrix} \bar{\lambda}_1 + \sigma^2 & & & & & \\ & \ddots & & & & \\ & & \bar{\lambda}_N + \sigma^2 & & & \\ & & & \sigma^2 & & \\ & & & & \ddots & \\ & & & & & \sigma^2 \end{pmatrix}
\end{aligned}$$

By writing the eigenvalues  $\lambda_k$  of  $\underline{\underline{\Lambda}}$  in a more compact form we get

$$\lambda_k = \begin{cases} \bar{\lambda}_k + \sigma^2, & \text{if } 1 \leq k \leq N \\ \sigma^2, & \text{if } N+1 \leq k \leq M \end{cases}$$

Roughly speaking, we can say that since

$$\underline{\underline{R}} = \underline{\underline{M}} \underline{\underline{\Lambda}} \underline{\underline{M}}^H = (\underline{\underline{U}} \underline{\underline{V}}) \begin{pmatrix} \bar{\lambda}_1 + \sigma^2 & & & & & \\ & \ddots & & & & \\ & & \bar{\lambda}_N + \sigma^2 & & & \\ & & & \sigma^2 & & \\ & & & & \ddots & \\ & & & & & \sigma^2 \end{pmatrix} \begin{pmatrix} \underline{\underline{U}}^H \\ \underline{\underline{V}}^H \end{pmatrix}$$

- The eigenvectors in  $\underline{\underline{U}}$  are associated to the source signals;
- The eigenvectors in  $\underline{\underline{V}}$  are associated to the noise.

Of course, this is valid just if the noise is lower than the sound coming from the source because  $\bar{\lambda}_k$  must dominate  $\sigma^2$  so as we can say that the eigenvectors in  $\underline{\underline{U}}$  are associated to the source signals and not to the noise.

Let us recap the condition below which this matrix can be obtained:

1. The number of sources  $N$  is known and we have more microphones than sources  $N < M$ ;
2. The noise power is lower than the source strengths.

All this to say that the parametric methods are based on some properties of  $\underline{\underline{U}}$  and  $\underline{\underline{V}}$ .

Let us see which are these properties:

1. *Property of the noise eigenvector  $\underline{\underline{V}}$*

The eigenvector decomposition ( $\underline{\underline{A}} \underline{\underline{v}} = \lambda \underline{\underline{v}} \rightarrow \underline{\underline{A}} \underline{\underline{V}} = \underline{\underline{V}} \underline{\underline{\Lambda}}$ ) implies that

$$\underline{\underline{R}} \underline{\underline{V}} = \underline{\underline{V}} \begin{pmatrix} \lambda_{N+1} & & \\ & \ddots & \\ & & \lambda_M \end{pmatrix} = \underline{\underline{V}} \begin{pmatrix} \sigma^2 & & \\ & \ddots & \\ & & \sigma^2 \end{pmatrix} = \sigma^2 \underline{\underline{V}}$$

Replacing the expression of  $\underline{\underline{R}}$ , we obtain

$$\begin{aligned} \underline{\underline{R}} \underline{\underline{V}} &= \underline{\underline{V}} \left( \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H + \sigma^2 \underline{\underline{I}}_M \right) \underline{\underline{V}} = \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{V}} + \sigma^2 \underline{\underline{I}}_M \underline{\underline{V}} = \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{V}} + \sigma^2 \underline{\underline{V}} = \\ &\stackrel{\substack{\underline{\underline{R}} = \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H + \sigma^2 \underline{\underline{I}}_M}{\underline{\underline{R}} \underline{\underline{V}} = \sigma^2 \underline{\underline{V}}}}{=} \sigma^2 \underline{\underline{V}} \rightarrow \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{V}} + \sigma^2 \underline{\underline{V}} = \sigma^2 \underline{\underline{V}} \xrightarrow{\substack{\text{That is true} \\ \text{if and only if}}} \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{V}} = \underline{\underline{0}}_{M \times (M-N)} \end{aligned}$$

The equation above, in turns, is verified if and only if

$$\underline{\underline{A}}^H \underline{\underline{V}} = \underline{\underline{0}}_{M \times (M-N)}$$

Since we cannot control  $\underline{\underline{R}_s}$  being the covariance of the sources (greater than zero by definition). From that we conclude that the null-space of our propagation vectors will be given by

$$\underline{\underline{A}}^H \underline{\underline{V}}$$

2. *Property of the signal eigenvector  $\underline{\underline{U}}$*

The eigenvector decomposition ( $\underline{\underline{A}} \underline{\underline{v}} = \lambda \underline{\underline{v}}$ ) implies that

$$\begin{aligned} \underline{\underline{R}} \underline{\underline{U}} &= \underline{\underline{U}} \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{pmatrix} \stackrel{\substack{\underline{\underline{\lambda}}_k = \begin{cases} \bar{\lambda}_k + \sigma^2, & \text{if } 1 \leq k \leq N \\ \sigma^2, & \text{if } N+1 \leq k \leq M \end{cases}}}{=} \underline{\underline{U}} \begin{pmatrix} \bar{\lambda}_1 + \sigma^2 & & \\ & \ddots & \\ & & \bar{\lambda}_N + \sigma^2 \end{pmatrix} = \\ &= \underline{\underline{U}} \begin{pmatrix} \bar{\lambda}_1 & & \\ & \ddots & \\ & & \bar{\lambda}_N \end{pmatrix} + \underline{\underline{U}} \begin{pmatrix} \sigma^2 & & \\ & \ddots & \\ & & \sigma^2 \end{pmatrix} \stackrel{\substack{\underline{\underline{\Lambda}} = \begin{pmatrix} \bar{\lambda}_1 & & \\ & \ddots & \\ & & \bar{\lambda}_N \end{pmatrix}}}{=} \underline{\underline{U}} \underline{\underline{\Lambda}} + \sigma^2 \underline{\underline{U}} \end{aligned}$$

Replacing the expression of  $\underline{\underline{R}}$ , we obtain

$$\begin{aligned} \underline{\underline{R}} \underline{\underline{U}} &= \underline{\underline{U}} \left( \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H + \sigma^2 \underline{\underline{I}}_M \right) \underline{\underline{U}} = \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{U}} + \sigma^2 \underline{\underline{I}}_M \underline{\underline{U}} = \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{U}} + \sigma^2 \underline{\underline{U}} = \\ &\stackrel{\substack{\underline{\underline{R}} \underline{\underline{U}} = \sigma^2 \underline{\underline{V}}}}{=} \underline{\underline{U}} \underline{\underline{\Lambda}} + \sigma^2 \underline{\underline{U}} \rightarrow \\ &\rightarrow \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{U}} + \sigma^2 \underline{\underline{U}} = \underline{\underline{U}} \underline{\underline{\Lambda}} + \sigma^2 \underline{\underline{U}} \rightarrow \underline{\underline{U}} \underline{\underline{\Lambda}} = \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{U}} \end{aligned}$$

We notice that  $\underline{\Lambda}$  is not singular

Why is  $\underline{\Lambda}$  not singular?

Reminder: the determinant of a diagonal matrix is equal to the product of its diagonal elements. An eigenvalue is different from zero by definition, so the determinant of  $\underline{\Lambda}$ , the eigenvalue matrix, is surely different from zero. If the determinant is different from zero, the matrix is invertible, thus it is not singular.

To keep this in mind, we just have to remind that when we evaluate the inverse of a matrix, we have to compute a division by its determinant.

$$A^{-1} = \frac{\text{adj } A}{\det A}$$

Going back to our property, we were saying that noticing that  $\underline{\Lambda}$  is not singular, we can invert it writing that

$$\underline{\underline{U}} = \underline{\underline{A}} \underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{U}} \left( \underline{\underline{\Lambda}} \right)^{-1}$$

Let  $\underline{\underline{C}} = \underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{U}} \left( \underline{\underline{\Lambda}} \right)^{-1}$

Where  $\underline{C}$  is a non-singular matrix whose elements are associated to

- The covariance matrix of the sources  $\underline{\underline{R}}_s$
  - The propagation matrix  $\underline{\underline{A}}^H$
  - “The eigenspace of the covariance matrix transformed in the space of the propagation matrix  $\underline{\underline{U}} \left( \underline{\underline{\Lambda}} \right)^{-1}$ ”

Let us see now which are these so-called parametric methods.

### *MUSIC (MUltiple SIgnal Classification)*

This is the first method we see. It exploits the property of noise eigenvectors that is

$$\underline{\underline{A}}^H \underline{\underline{V}} = \underline{\underline{0}}_{M \times (M-N)}$$

The above expression can be written in a scalar form (minimising the norm of  $\underline{\underline{0}}_{M \times (M-N)}$ ). It becomes

$$\underline{\underline{A}}^H \underline{\underline{V}} \underline{\underline{V}}^H \underline{\underline{A}} = 0$$

We have to find a solution to this equation. We remind that we do not know  $\underline{\underline{A}}$  (the propagation matrix).  $\underline{\underline{A}}$  is the unknown in our equation. We have to find the vectors  $\underline{\underline{a}}(\theta_k)$  that verifies the equation below. They will be the one for which the  $\theta$  is exactly the same of DOA. Mathematically speaking, we have to find those  $\theta_k$  for which

$$\underline{\underline{a}}^H(\theta_k) \underline{\underline{V}} \underline{\underline{V}}^H \underline{\underline{a}}(\theta_k) = 0 \quad \text{for } k = 1, \dots, N$$

The idea is always the same: we test several values of  $\theta$  (DOAs) that correspond to different propagation vectors and we test which of them satisfy the condition written above. They will be the one for which a peak in the pseudospectrum appears.

How do we do it? By following the passages illustrated in the MUSIC algorithm:

- a. Estimate the covariance matrix

$$\hat{\underline{\underline{R}}} = \frac{1}{K} \sum_{t=1}^K \underline{\underline{y}}(t) \underline{\underline{y}}^H(t)$$

- b. From the eigenvalues decomposition (under all the assumptions we saw in the last lecture), estimate the eigenvectors related to noise  $\hat{\underline{\underline{V}}}$
- c. Estimate the DOAs as the  $N$  highest peaks of the function:

$$p(\theta) = \frac{1}{\underline{\underline{a}}^H(\theta_k) \hat{\underline{\underline{V}}} \hat{\underline{\underline{V}}}^H \underline{\underline{a}}(\theta_k)}, \quad \theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$$

When we vary  $\underline{\underline{a}}(\theta_k)$  (according to  $\theta$ ), the quantity at the denominator goes to zero. (Actually, not exactly zero due to the huge number of approximations we did. Furthermore, the propagation vector itself is an ideal one).

What we get is a great pseudospectrum with a higher resolution with respect to spatial filtering methods but at the cost of computing the eigenvalue decomposition and having a large number of assumptions verified. At the end, this technique is not so far from the pseudospectrum evaluation made with spatial frequencies.

Pros	Cons
High resolution	Computationally heavy (eigendecomposition) Large number of assumptions to verify

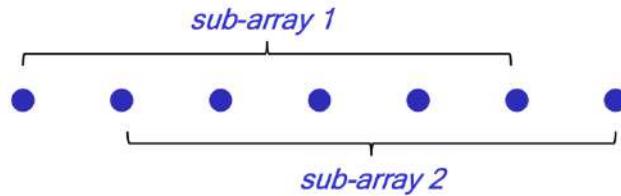
### ESPRIT (Estimation of Signal Parameters via Rotational Invariance Technique)

The ESPRIT method is completely different from the method we have seen up to now. In this case we do not guess the spatial frequency from the peaks of pseudospectrum, we estimate them directly from the signal. This technique is based upon the properties of the signal eigenvectors  $\underline{\underline{U}}$ :

$$\underline{\underline{U}} = \underline{\underline{A}} \underbrace{\underline{\underline{R}_s} \underline{\underline{A}}^H \underline{\underline{U}}}_{\underline{\underline{C}}} \left( \underline{\underline{\Lambda}} \right)^{-1}$$

Let us go through it! The idea is really smart: we try to compute the spatial frequencies just by relying on the fact that this information is encoded in the delay between the signals of different microphones.

Let us consider that we have an  $M$  microphones array that we split in two identical sub-arrays, each containing  $\bar{M} = M - 1$  elements



We can select the two sub arrays by means of a selection matrix that is nothing else but an identity matrix with a zero replacing the last one of the diagonal when we want to activate the sub-array 1 and a zero that replaces the first one of the diagonal when we want to activate the sub-array 2.

In light of this fact, we can define the signals  $\underline{y}_1(t)$  and  $\underline{y}_2(t)$  by taking only the propagation vectors of microphones that are activated in the two cases.

So, the array data model in general is  $\underline{y}(t) = \underline{\underline{A}} \underline{s}(t) + \underline{e}(t)$ , from which we derive the subarray models

$$\begin{cases} \underline{y}_1(t) = \underline{\underline{A}}_1 \underline{s}(t) + \underline{e}_1(t), & \underline{\underline{A}}_1 = \left( \underline{\underline{I}}_{\bar{M}} \underline{\underline{0}} \right) \underline{\underline{A}} \\ \underline{y}_2(t) = \underline{\underline{A}}_2 \underline{s}(t) + \underline{e}_2(t), & \underline{\underline{A}}_2 = \left( \underline{\underline{0}} \underline{\underline{I}}_{\bar{M}} \right) \underline{\underline{A}} \end{cases}$$

Knowing that, we can observe that the two propagation vectors are the same but shifted by a complex exponential. Since they are shifted just of a distance of one microphone  $\tau = 1$ . What we get is

$$\underline{\underline{A}}_2 = \underline{\underline{A}}_1 \underline{\underline{D}} \quad \text{where } \underline{\underline{D}} = \begin{pmatrix} e^{-j\omega_{s_1}} & & \\ & \ddots & \\ & & e^{-j\omega_{s_N}} \end{pmatrix}$$

The idea, at the end, is to retrieve the spatial frequency from the shift between the two sub-arrays which are directly related to the source DOAs. Let us try to estimate them.

Everything boils down to the eigenvalue decomposition of the spatial covariance matrix of the two sub-arrays knowing that we have this special relation between the two sub-arrays. From their eigenvalues, we can compute the eigenvectors associated to the sources, namely  $\underline{U}_1$  and  $\underline{U}_2$ , which can also be computed as:

$$\begin{cases} \underline{\underline{U}}_1 = \left( \underline{\underline{I}}_{\bar{M}} \underline{\underline{0}} \right) \underline{\underline{U}} \\ \underline{\underline{U}}_2 = \left( \underline{\underline{0}} \underline{\underline{I}}_{\bar{M}} \right) \underline{\underline{U}} \end{cases}$$

Now that we know this relation, we can write  $\underline{\underline{U}}$ , the eigenvector matrix of the source as the product between  $\underline{\underline{A}}$ , the propagation matrix, and  $\underline{\underline{C}}$ , an invertible transformation matrix. From this assumption, it follows that

$$\begin{aligned} \underline{\underline{U}}_1 &= (\underline{\underline{I}}_{\bar{M}} \underline{\underline{0}}) \underline{\underline{U}} = \underline{\underline{U}} \stackrel{\underline{\underline{U}} = \underline{\underline{A}} \underline{\underline{C}}}{=} (\underline{\underline{I}}_{\bar{M}} \underline{\underline{0}}) \underline{\underline{A}} \underline{\underline{C}} = \underline{\underline{A}}_1 \underline{\underline{C}} \\ \underline{\underline{U}}_2 &= (\underline{\underline{0}} \underline{\underline{I}}_{\bar{M}}) \underline{\underline{U}} = \underline{\underline{U}} \stackrel{\underline{\underline{U}} = \underline{\underline{A}} \underline{\underline{C}}}{=} (\underline{\underline{0}} \underline{\underline{I}}_{\bar{M}}) \underline{\underline{A}} \underline{\underline{C}} = \underline{\underline{A}}_2 \underline{\underline{C}} \end{aligned} \quad \begin{array}{l} \xrightarrow{\text{We selected the propagation matrix for } \underline{\underline{A}}_1 \underline{\underline{C}} \text{ and } \underline{\underline{A}}_2 \underline{\underline{C}}} \quad \begin{cases} \underline{\underline{A}}_1 = \underline{\underline{U}}_1 \underline{\underline{C}}^{-1} \\ \underline{\underline{A}}_2 = \underline{\underline{U}}_2 \underline{\underline{C}}^{-1} \end{cases} \\ \xrightarrow{\underline{\underline{A}}_2 = \underline{\underline{A}}_1 \underline{\underline{D}}} \quad \begin{cases} \underline{\underline{A}}_1 = \underline{\underline{U}}_1 \underline{\underline{C}}^{-1} \\ \underline{\underline{A}}_1 \underline{\underline{D}} = \underline{\underline{U}}_2 \underline{\underline{C}}^{-1} \end{cases} \rightarrow \underline{\underline{U}}_1 \underline{\underline{C}}^{-1} \underline{\underline{D}} = \underline{\underline{U}}_2 \underline{\underline{C}}^{-1} \rightarrow \underline{\underline{U}}_2 = \underline{\underline{U}}_1 \underline{\underline{C}}^{-1} \underline{\underline{D}} \underline{\underline{C}} \end{array} \quad \begin{array}{l} \underline{\underline{U}}_2 = \underline{\underline{U}}_1 \underline{\underline{\Phi}} \\ \text{Let } \underline{\underline{\Phi}} = \underline{\underline{C}}^{-1} \underline{\underline{D}} \underline{\underline{C}} \end{array}$$

Since  $\underline{\underline{\Phi}}$  and  $\underline{\underline{D}}$  are similar, they will have the same eigenvalues. Our goal is to estimate it. Pay attention to the fact that  $\underline{\underline{\Phi}}$  is not a square matrix: this means that we cannot compute its inverse but its pseudoinverse.

Reminder: generalised matrix

Given an  $m \times n$  matrix  $A$ , an  $n \times m$  matrix  $G$  is said to be a generalised inverse of  $A$  if  $AGA = A$ .

Let us see how to find these eigenvalues:

- Firstly, we compute the sample covariance matrix  $\widehat{\underline{\underline{R}}}$
- From the eigenvalue decomposition of  $\widehat{\underline{\underline{R}}}$ , we compute  $\widehat{\underline{\underline{U}}}_1$  and  $\widehat{\underline{\underline{U}}}_2$  by selecting the eigenvectors of the first and the second sub-arrays,
- Then, we estimate  $\widehat{\underline{\underline{\Phi}}}$  as pseudoinverse of  $\widehat{\underline{\underline{U}}}_1$ :  $\widehat{\underline{\underline{\Phi}}} = (\widehat{\underline{\underline{U}}}_1^H \widehat{\underline{\underline{U}}}_1)^{-1} \widehat{\underline{\underline{U}}}_1^H \widehat{\underline{\underline{U}}}_2$
- We compute the eigenvalues of  $\widehat{\underline{\underline{\Phi}}}$ :  $v_1, \dots, v_N$  that will coincide with the ones of  $\widehat{\underline{\underline{D}}}$
- Looking at the argument of these complex exponential, we get the spatial frequencies

$$\widehat{\underline{\underline{D}}} = \begin{pmatrix} v_1 = e^{-j\omega_{s_1}} & & \\ & \ddots & \\ & & v_N = e^{-j\omega_{s_N}} \end{pmatrix} \quad \omega_{s_k} = -\arg(v_k), \quad k = 1, \dots, N$$

- At the end, we can infer the direction of arrival using the relation

$$\omega_s \triangleq \omega_c \frac{d \sin(\theta)}{c}$$

As we can notice this technique is very powerful since we do not have to compute a complex function (the pseudospectrum one) for multiple  $\theta$ , making discretisation etc... In this case we estimate directly the spatial frequency. Furthermore, the resolution is higher since it does not depend on the discretisation choice we have to make for the spatial domain.

Pros	Cons
High resolution (it does not depend on the discretisation choice)	We need an array that can be divided in sub-arrays (two microphones are not enough)
Powerful since we estimate directly the spatial frequencies	It requires the computation of eigenvalue decomposition

## Localisation of wide-band sources

All these techniques we saw up to now are based on a signal model that is narrow band, which means that, if we have a broadband signal (so every signal that is not a sinusoid), the method needs to be applied frequency by frequency. Of course, these methods are inefficient when we have to deal with broadband signals. That is why we come up with these new classes of methods that are based not on frequency decomposition but on time domain. Also, the set-up is slightly different: from now on, we consider only one source but, this time we are neither in free-field nor in far field. In fact, we consider small/moderate reverberations with whose we have to deal. The two techniques we will see for source-localisation with broadband signals are:

- Generalised Cross Correlation
- Adaptive Eigenvalue Decomposition

## Data Model

Since we work with wide band sources, we cannot use the narrow band model, so we work directly in time domain. All the considerations will be done with a microphone array made by two microphones that is, by the way, the most common situation. Every device has at most two microphones since the more microphones we have, the higher the cost of the device. The aim of the companies is always to reduce the costs.

The wide-band array model can be expressed as

$$y_k(n) = g_k(n) * s(n) + e_k(n), \quad k = 1, 2$$

Where:

- $s(n)$  is the source signal
- $g_k(n)$  is the channel response from the source to the microphone
- $e_k(n)$  is the additive noise at the microphone
- $y_k(n)$  is the microphone signal

### *Time Difference of Arrival (TDOA)*

The room impulse response is also called the channel response since it is something that stands in between a source and a receiver, as in telecommunications system a channel is defined. In this case, it can be modelled with a set of Dirac pulses: some of them will be the one related to the direct sound while the others will be the one related to the reflections.

$$g_1(n) = \underbrace{\frac{1}{r_1} \delta(n - \tau_1)}_{\text{Direct Signal}} + \underbrace{\sum_i \alpha_i \delta(n - \tau_{1i})}_{\text{Early reflections}}$$

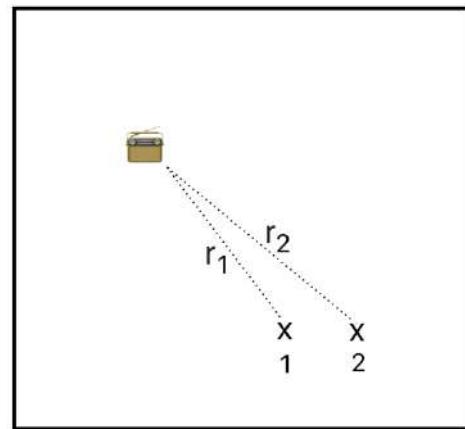
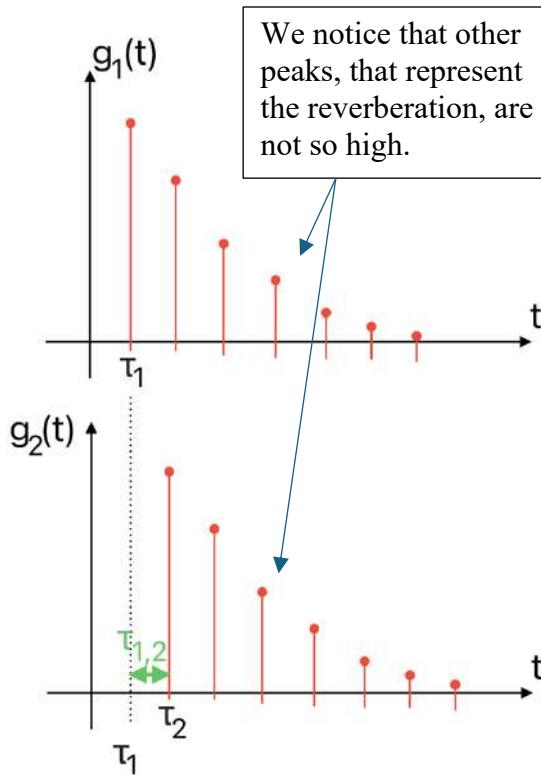
$$g_2(n) = \underbrace{\frac{1}{r_2} \delta(n - \tau_2)}_{\text{Direct Signal}} + \underbrace{\sum_i \beta_i \delta(n - \tau_{2i})}_{\text{Early reflections}}$$

According to where the source is located, the direct path (and consequently the reflection) will arrive to the microphone with different delays. So, the information about the position, if we neglect the reflections, will be encoded in the relative delay between the microphone as it happens in narrow-band model. This difference is called Time Difference of Arrival (TDOA), a.k.a. time delay and it is define by:

$$\tau_{1,2} \triangleq \tau_2 - \tau_1$$

Let us dwell on this concept with an example to understand which is the difference between the narrow-band signal model and the wide-band one. We suppose to have two channels response  $g_1(t)$  and  $g_2(t)$ . In the first one, the first peak will arrive at a certain time  $\tau_1$  while in the second one at a time  $\tau_2$ . Of course, these

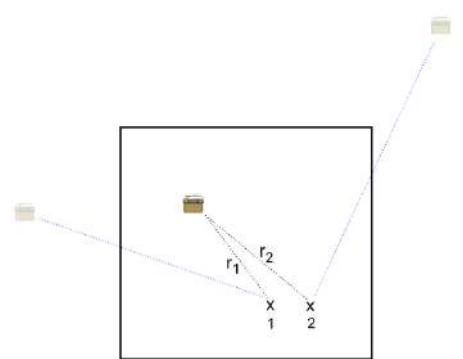
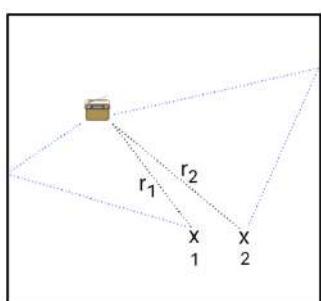
time values  $\tau_1$  and  $\tau_2$  will be related to the position of the sources since they represent the time that the signal needs to travel from the source to the microphone. In light of this fact, since  $\tau_2 > \tau_1$  we can imagine that the second microphone will be farther than the first one. The front wave will arrive first to microphone 1 than to the second one. A possible scenario could be the following.



Unfortunately, we do not know this information. We do not have neither the channel response nor the absolute time instance  $\tau_1$  and  $\tau_2$ . We just receive a couple of signals that are the convolution between the source signal and the two channel responses.

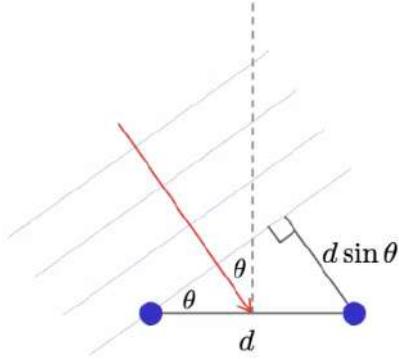
Then we have the reflections (in the example are reported in blue just two rays but we have to imagine that the rays are infinite and spread all around the source).

Since it is not that easy to analyse paths that are not represented by a straight line, we usually apply the so-called “mirroring sources technique” that allows us to represent the path of the reflected sound with a straight line coming from a “copy” of the source. We simply transpose the line that goes from the wall to the source, along the line that goes from the microphone to the wall.



So, each time delay  $\tau_{1,i}$  and  $\tau_{2,i}$  will express the time that the sound takes to travel from a mirrored source to the microphone. We notice that in  $g_1(t)$  and  $g_2(t)$  the spikes related to the reflections are lower than the one related to the direct signal. This is because the longer the path, the higher is the energy losses. Furthermore, a not negligible amount of energy is lost in the reflections, and it transferred to the walls. Therefore, we can conclude that the delays are related to the direction of arrival of the signal from the source.

But how can we relate this delay (TDOA) to the DOA? Assuming that the source is in the far-field with respect to the microphone pair (which implies that we can approximate the spherical front wave with parallel ones), the TDOA is related to the DOA through a simple trigonometric relationship:



$$\tau_{1,2} = \frac{d \sin(\theta)}{c}$$

Given an estimate of the TDOA  $\hat{\tau}_{1,2}$ , the DOA is computed as:

$$\hat{\theta} = \arcsin\left(\frac{\hat{\tau}_{1,2} c}{d}\right)$$

In light of this, the problem of DOA estimation can be brought to the problem of estimating the TDOA. Now that we have our data model, our goal is to estimate the TDOA. In order to do so, we will see two techniques.

- A first approach, more intuitive, that is based on looking for the maxima of the cross-correlation of the microphone signals called Generalised cross-correlation method;
- A second and more advanced approach in which we identify directly the two channel responses  $g_1$  and  $g_2$  called Adaptive Eigenvalue Decomposition (AED) method.

### Generalised cross-correlation (GCC) method

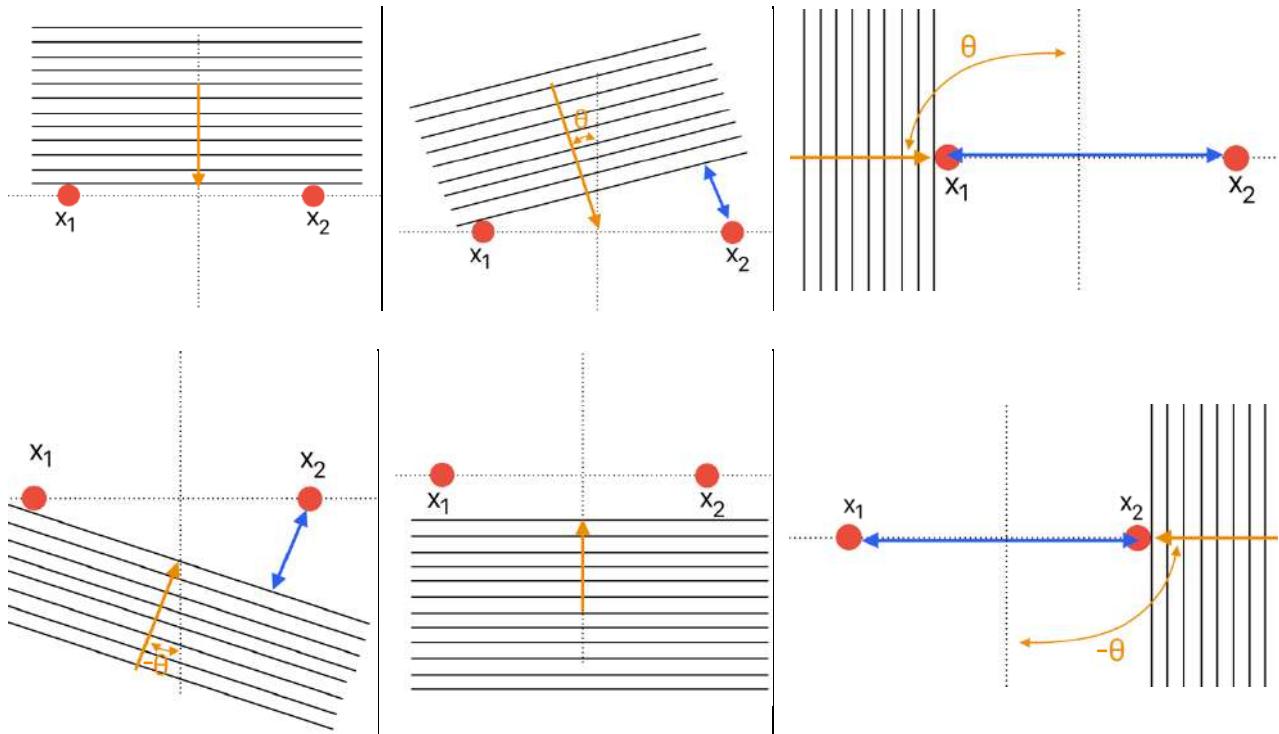
The two microphones are expected to record two delayed copies of the source signal, along with some disturbances such as the environmental noise and the reverberation. If the amount of noise and reverberation is low compared to the strength of the source signal, we can cross-correlate the two microphone signals to estimate the TDOA where the cross-correlation of the microphone signals will be given by the inverse Fourier Transform of the cross-spectrum (power of the correlation):

$$\begin{aligned} R_{1,2}(\tau) &\triangleq E\{x_1(t)x_2(t)\} \stackrel{\substack{\omega \\ \text{Wiener--Хинчин}}}{=} \mathcal{F}^{-1}\{S_{1,2}(\omega)\} = \\ &= \frac{1}{2\pi} \int_{-\infty}^{\infty} S_{1,2}(\omega) e^{j\omega\tau} d\omega \stackrel{\substack{\approx \\ S_{1,2}(\omega) \triangleq E\left\{\lim_{T \rightarrow \infty} \frac{|X_1(\omega)X_2^*(\omega)|^2}{T}\right\}}}{\approx} \frac{1}{2\pi} \int_{-\infty}^{\infty} X_1(\omega)X_2^*(\omega) e^{j\omega\tau} d\omega \end{aligned}$$

The cross-correlation between the signals of the two microphones will be maximised when the lag-time  $\tau$  will be equal to the delay between them ( $\hat{\tau}_{1,2}$ ) since all the peaks of the function will be aligned maximising the product. Formally

$$\hat{\tau}_{1,2} = \arg \max_{\tau \in \mathcal{D}} R_{1,2}(\tau) \quad \text{where } \mathcal{D} = \left[-\frac{d}{c}, \frac{d}{c}\right]$$

We remind that  $d$  is the distance between the two microphones and  $c$  is the speed of sound. Why do we choose these boundaries? Surely the signal is long, and it makes no sense to consider every possible time lag. We already know that the delay has an upper boundary: since it depends on the distance, we can observe the maximum (in absolute value) delay when the signal is propagating parallel to the line on which the two microphones are placed. Let us try to understand this concept with the following small drawing.



As we can see, when  $\theta = 0^\circ$  the delay between the microphones will be maximum and the maximum delay will be given by  $\theta = \pm 90^\circ$  so when

$$\max \tau_{1,2} = \frac{d \sin(\pm 90^\circ)}{c} = \pm \frac{d}{c}$$

And that is why  $\mathcal{D} = \left[ -\frac{d}{c}, \frac{d}{c} \right]$ .

This is very nice, but we can have some problems. In particular, when the signal presents some periodicity, so each time we have an harmonic signal, typical condition when we deal with sound. In this case also the cross correlation will be periodic. Since the cross-correlation function will present peaks that arise from the auto-correlation of the signal, being this signal periodic, there will not be a clear single peak. How can we solve this problem?

A very smart observation can be done. We want to have an autocorrelation that is impulsive in order to have just one single peak in the function being able to identify easily the peak related to the delay. Which is the signal that have an autocorrelation with a single peak? The white noise! So, the idea is to resemble the cross-correlation of white noise. Again, we can exploit the Wiener-Hinchin theorem.

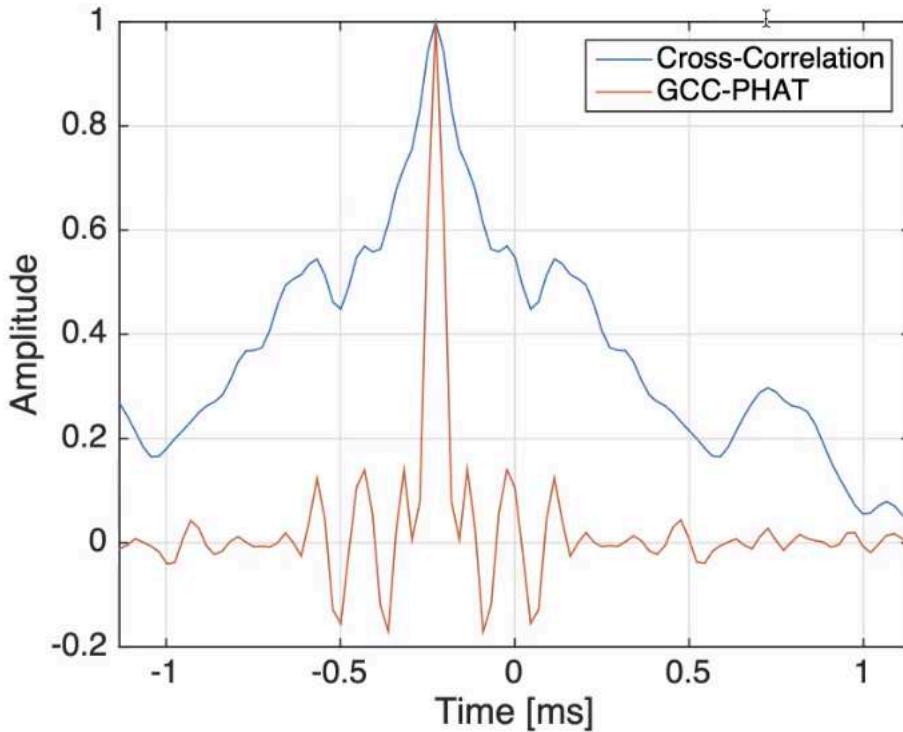
The autocorrelation of white noise presents a cross spectrum that is flat. In order make the signal white, we have to find a way to flat the power spectrum. How can we obtain it? By dividing the cross-spectrum by itself. In this way, the spectrum is flat, and the autocorrelation presents just a spike in zero. However, to have a complete flat spectrum, we would need a signal with infinite frequencies. This is not possible, so we have to truncate the frequency content of our signal. This will generate ringing in time domain.

We formally generalise the cross correlation of the signal artificially whitened with the following formula

$$R_{1,2}(\tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \psi_{1,2}(\omega) X_1(\omega) X_2^*(\omega) e^{j\omega\tau} d\omega$$

Where  $\psi_{1,2}(\omega) = \frac{1}{|X_1(\omega)X_2^*(\omega)|}$  is a weight function (that is there to divide the spectrum for itself) called PHAT (PHASE Taransform). As always, nomen omen: the magnitude is equalised to 1 independently from the function while the phase still depends on it.

Let us see the effects of this PHAT function on the spectrum with an example.

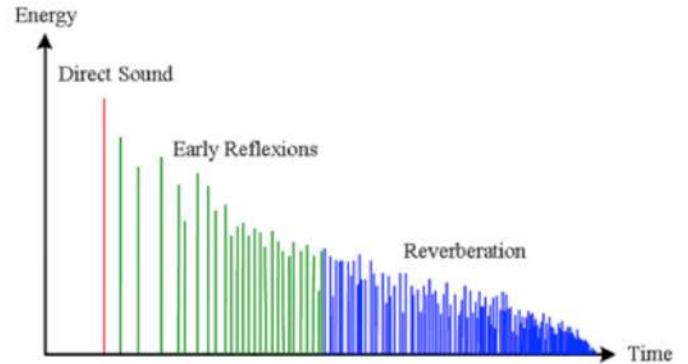


As we can see, after the phase transform, the peak of the cross-correlation function looks sharper than before. We can also notice the small amount of ringing around the peak due to the fact that we are not considering infinite frequencies.

### Adaptive Eigenvalue Decomposition (AED) method

AED aims at estimating directly the channel responses, from which it is possible to estimate the TDOA.

As we can intuitively observe, the first peak in the impulse responses is related to the direct-path from the source to the microphones since the direct path is the shortest at the same time. Being the speed of sound the same, the signal that is propagating along the direct-path will be the one that will be received firstly.



We can recover the TDOA as the difference between the time-lags related to the direct-paths at the two microphones.

The estimation of the channel starts from an interesting observation. If we are in a NON-noisy scenario (condition in which the SNR is high for example when we work with very good microphones, or we can discard the noisy components) we can tell that

$$\underline{y}_1(n) * \underline{g}_2(n) = s(n) * \underline{g}_1(n) * \underline{g}_2(n) \quad \stackrel{\text{linear operation}}{=} \quad \underline{y}_2(n) * \underline{g}_1(n)$$

$$\underline{y}_2(n) = s(n) * \underline{g}_2(n)$$

This expression can be approximated considering M-long frames of the microphone signal and expressed in vectorial form

$$\underline{y}_1^T(n) \underline{g}_2 = \underline{y}_2^T(n) \underline{g}_1 \quad \Rightarrow \quad \underline{y}_1^T(n) \underline{g}_2 - \underline{y}_2^T(n) \underline{g}_1 = 0$$

Where:

$$\underline{y}_k(n) = [y_k(n), y_k(n-1), \dots, y_k(n-M+1)]^T$$

$$\underline{g}_k = [g_k(0), g_k(1), \dots, g_k(M-1)]^T$$

We notice that we make the transpose instead of the Hermitian since we work with real signals.

We look for the vectors  $\underline{g}_1$  and  $\underline{g}_2$  that minimise the equation. In theory they should annul the product between them and the signals, but we know that these are just mathematical and ideal results that are based upon ideal assumption. Not wanting to be too pretentious, we try to find a solution in a statistical sense. We can consider the difference  $\underline{y}_1^T(n)\underline{g}_2 - \underline{y}_2^T(n)\underline{g}_1$  as an error (it is, in fact, the difference between two quantities) of which we want to minimise the energy (the quantity squared). By defining the covariance-matrix  $\underline{\underline{R}}$  and the channels vector  $\underline{\underline{u}}$

$$\underline{\underline{R}} = \begin{pmatrix} E\{\underline{y}_1(n)\underline{y}_1^T(n)\} & E\{\underline{y}_1(n)\underline{y}_2^T(n)\} \\ E\{\underline{y}_2(n)\underline{y}_1^T(n)\} & E\{\underline{y}_2(n)\underline{y}_2^T(n)\} \end{pmatrix} \quad \underline{\underline{u}} = \begin{pmatrix} \underline{g}_2 \\ -\underline{g}_1 \end{pmatrix}$$

we can express the minimisation problem as

$$\underline{\underline{R}} \underline{\underline{u}} = \underline{\underline{0}}_{2M}$$

This tells us that  $\underline{\underline{u}}$  is the Null-space of  $\underline{\underline{R}}$  that means that  $\underline{\underline{u}}$  is exactly the eigenvector associated to the zero-valued eigenvalue of  $\underline{\underline{R}}$ . Furthermore, if the two impulse responses have no common zeros, there will be just one eigenvalue that verifies this equation since  $\underline{g}_1$  and  $\underline{g}_2$  will not be linearly related.

Of course, this  $\underline{\underline{R}}$  matrix is derived for an ideal situation in which we are able to know exactly the expected value of the products of the signals. As we can imagine, in reality, we can rely only on approximation of  $\underline{\underline{R}}$  and so we will not find exactly zero-valued eigenvalues. What we will find is the minimum one. How can we retrieve it? As we did in the other methods: we look for the eigenvalues, we sort them according to their value and we pick the smallest one.

This is nice, but again we encounter some practical problems: the complexity is cubic in terms of the derivation of  $\underline{\underline{R}}$  (Since  $\underline{\underline{R}}$  is an  $M \times M$  matrix, the complexity will be  $O((2M)^3)$ ). Since we are working in time domain, with windows, we have several samples so  $M$  can become very high. This makes this method very inconvenient. In order to solve this problem, we come up with an idea: we do not try to properly minimise  $\underline{\underline{R}} \underline{\underline{u}}$  (operation for which we need to perform the eigenvalue decomposition), instead we try to minimise the norm of the vector  $\underline{\underline{R}} \underline{\underline{u}}$ . This ends up in solving the following constrained minimisation problem

$$\hat{\underline{\underline{u}}} = \arg \min_{\underline{\underline{u}}} \underline{\underline{u}}^T \underline{\underline{R}} \underline{\underline{u}} \quad \text{subject to } \underline{\underline{u}}^T \underline{\underline{u}} = 1$$

In order to solve this problem, we can use a LMS-like procedure (explained in the 19<sup>th</sup> March lecture). We briefly report the main passages:

1. We define the error signal as

$$e(n) = \frac{\underline{u}^T(n)\underline{y}(n)}{\|\underline{u}(n)\|} \quad \text{where } \underline{y}(n) = [\underline{y}_1^T \quad \underline{y}_2^T]^T$$

We notice that the MSE minimisation of  $e(n)$  corresponds to the constrained minimisation problem

2. We set of the LMS algorithm

$$\underline{u}(n+1) = \underline{u}(n) + \mu e(n) \nabla e(n)$$

Where:

$$\nabla e(n) = \frac{1}{\|\underline{u}(n)\|} \left[ \underline{y}(n) - e(n) \frac{\underline{u}(n)}{\|\underline{u}(n)\|} \right]$$

3. We iterate it.

4. After convergence, it can be proved that

$$R \frac{\underline{u}(\infty)}{\|\underline{u}(\infty)\|} = E\{e^2(\infty)\} \frac{\underline{u}(\infty)}{\|\underline{u}(\infty)\|}$$

That confirms that the iterative algorithms converges to the correct solution:  $E\{e^2(\infty)\}$  is the smallest eigenvalue of  $\underline{\underline{R}}$ .

Therefore, at the end we find two vectors ( $\hat{\underline{g}}_1$  and  $\hat{\underline{g}}_2$ , the one inside  $\underline{u}$ ) the condition  $\underline{\underline{R}} \underline{u} = \underline{0}_{2M}$  and that verify what we said at the beginning. We conclude that  $\hat{\underline{g}}_1$  and  $\hat{\underline{g}}_2$  (or, scalarly expressed  $\hat{g}_1(n)$  and  $\hat{g}_2(n)$ ) are correct approximation of our channels so of the impulse response of microphone one and microphone two. We do not have to do nothing but estimate the propagation delays by simply looking for the instant in which we have the maximum peak in the two functions,

$$\hat{\tau}_k = \arg \min_{\tau} \hat{g}_k(\tau) \quad \text{with } k = 1, 2$$

Finally, we estimate the TDOA as  $\tau_{1,2} = \hat{\tau}_2 - \hat{\tau}_1$

## References:

- “The generalized correlation method for estimation of time delay” C Knapp, G Carter - IEEE transactions on acoustics, speech and signal processing, 1974
- “Adaptive eigenvalue decomposition algorithm for passive acoustic source localization”, J Benesty - the Journal of the Acoustical Society of America, 2000

## Source Separation

Up to now we have seen some techniques to localise sources in space. Another important problem is the so-called source separation which means solving the cocktail problem. When you are at a cocktail party with people talking, we can speak to someone and understand what he is saying because we are able to separate its speech from the other sound in the environment. This is not easy at all to be done algorithmically. It represents one of the biggest problems in audio signal processing. Sometimes source separation is also used for mixing, when we want to extract a single part from a piece. Nowadays deep learning is the main tool used, but there were some interesting algorithmics used in the past that we will analyse.

Let us give a little bit of context by considering the presence of  $N$  sources in an environment, emitting the signals  $s_1(n), \dots, s_N(n)$ . We acquire the signal using  $M > 1$  microphones implementing a multichannel acquisition. The microphone signals  $y_1(n), \dots, y_M(n)$  will contain a mixture of the source signals. Our goal is to obtain an estimation of the source signals from the observed microphone signals: at the microphone we will have a so-called mixture of the sources. We want to estimate each single source at the microphone. According to the set up that we have, we can make different assumptions adopting different taxonomies to categorise the separation techniques used to tackle the problem:

1. The first question we have to ask to ourselves is how much information we have from about the mixing process.
2. Then if the environment reverberating or not
3. Another important question is how many sources we have in our scheme. Are the number of sources greater than the one of mic?

By answering to the first question, we can define two different sets in which to solve the problem:

- Blind Source Separation (BSS)

As we can imagine, it is the most challenging one since we have no information about the system. It is difficult also to infer information in this scenario. The only information that we have is the signal itself and some a-priori statistic on the signal, nothing about the single channels that compose the signal.

- Non blind source separation

We know roughly the impulse response (maybe we have determined with adaptive eigenvalue decomposition). The separation problem reduces to the inversion of a (possibly convolutive) system.

In our discussion, we will focus on the first one, that is the most common situation.

Another important thing that we have to take into account is the information about the environment (second question). As the environment change, also the information about the mixing process changes. According to the environment, the signal can be classified in two different categories:

- Instantaneous mixing

In this scenario we consider an environment with no (either very low) reverberation. Microphone signals can be modelled as a linear combinations of the source signals (possibly with complex-valued weights)

$$\begin{bmatrix} y_1(t) \\ \vdots \\ y_M(t) \end{bmatrix} = \begin{bmatrix} a_{1,1} & \cdots & a_{1,N} \\ \vdots & \ddots & \vdots \\ a_{M,1} & \cdots & a_{M,N} \end{bmatrix} \begin{bmatrix} s_1(t) \\ \vdots \\ s_N(t) \end{bmatrix}$$

Separation involves the inversion of the mixing matrix.

- Convulsive mixing

In this situation, we consider a reverberant environment. To separate the sources, we just have to invert the convulsive system:

$$y_k(n) = \sum_{i=1}^N \sum_{l=0}^{L-1} g_{ik}(l) s_i(n-l), \quad k = 1, \dots, M$$

An interesting trick we can exploit is that, if the room impulse response is short enough, by moving the signal from time domain to the bidimensional domain of the STFT a convulsive system becomes instantaneous since we analyse tracks cropped in little windows. This is not always true and that is why we consider low reverberations in our hypothesis.

Let us answer to the third question: is the number of microphones (M) greater than the number of sources (N)?

- If  $M > N$  we are talking about overdetermined separation problems.
- If  $M < N$  we are dealing with underdetermined separation problems. In this second situation we have to estimate the mixing model using some a-priori knowledge of the information model.

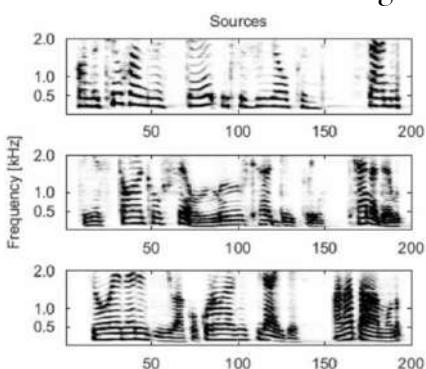
### Source separation using binary masks

We see how to use binary mask to separate acoustic sources. As we will see, we will rely on the assumption that in the STFT domain the speech signals are sparse.

Let us consider the most difficult scenario:

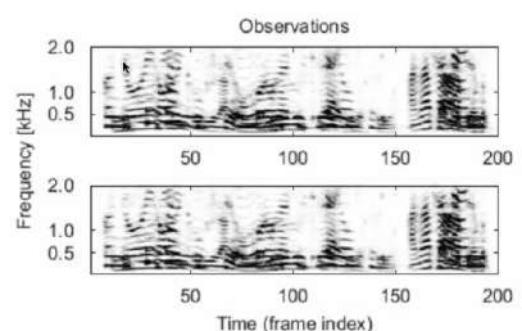
- o Undetermined problem:  $M = 2$  microphones,  $N = 3$  sources.
- o Blind Source Separation
- o Low-reverberation
- o Sources fixed in space, speech signals (a speech signal is generally sparse)

We will work in this domain because of the sparsity, and we will perform source separation by means of masking of the spectrogram. A mask is function multiplied elementwise in the STFT domain in order to extract information from the signal (in our case, in order to extract the track).



On the left we have the STFT of the three sources. Each point of this picture corresponds to one position of the time window and to one frequency band. There are many points that are white (no energy associated to the corresponding frequency at that time instant) and others that are black (high energy). This confirms that the signal is sparse in this domain as we do not have constant values everywhere: sometimes we have information and sometimes not. How can we say that these are speech signals? Because of the formants (on the frequency axis) that evolve in time. Then we

have the two observations: the mixed signal acquired by the microphones. Algebraically speaking, they are nothing but linear combinations of the signals produced by the sources. Furthermore, having assumed that the signals in the STFT domain are sparse (ideally where one is white the other one is black), we can affirm that for each time-frequency pair the sound will be associated only to one source.



Given the fact that they are sparse (so, basically, they do not emit the same signals), they will be also independent. Let us define:

- $Y_1(n, \omega_k), Y_2(n, \omega_k)$  as the STFTs of microphone signals
- $S_1(n, \omega_k), S_2(n, \omega_k), S_3(n, \omega_k)$  as the STFTs of source signals

Our goal is to find 3 binary masks  $M_1, M_2, M_3$  such that the source STFTs can be estimated as:

$$\hat{S}_i(n, \omega_k) = Y_i(n, \omega_k) \cdot M_i(n, \omega_k) \quad i = 1, 2, 3$$

This technique is also called hard masking since our mask is composed only by binary values (0 and 1).

The point is, how can we estimate the mask. Since we have a multichannel acquisition, we can exploit spatial information. In absence of reverberation, microphone signals will contain attenuated and delayed version of the source signals. Example for bins associated to the  $i$ -th source we will have:

$$\begin{cases} Y_1(n, \omega_k) = \alpha_{1,i} \cdot S_i(n, \omega_k) e^{-j\omega_k \tau_{1,i}} \\ Y_2(n, \omega_k) = \alpha_{2,i} S_i(n, \omega_k) e^{-j\omega_k \tau_{2,i}} \end{cases}$$

Each time-frequency bin associated to the  $i$ -th source will exhibit:

- Constant time-delay:  $T_i(n, \omega_k) = \tau_{i,2} - \tau_{i,1}$
- Constant attenuation ratio:  $A_i = \frac{\alpha_{2,i}}{\alpha_{1,i}}$

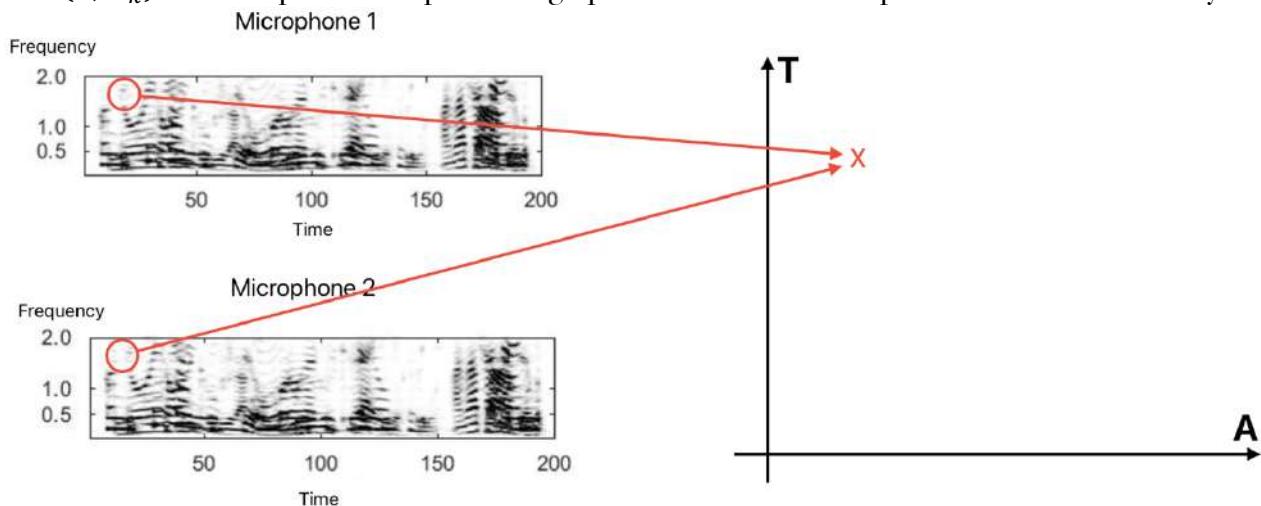
Since the source are not moving, the delay and the attenuation will remain the same: constant over time.

### Methodology

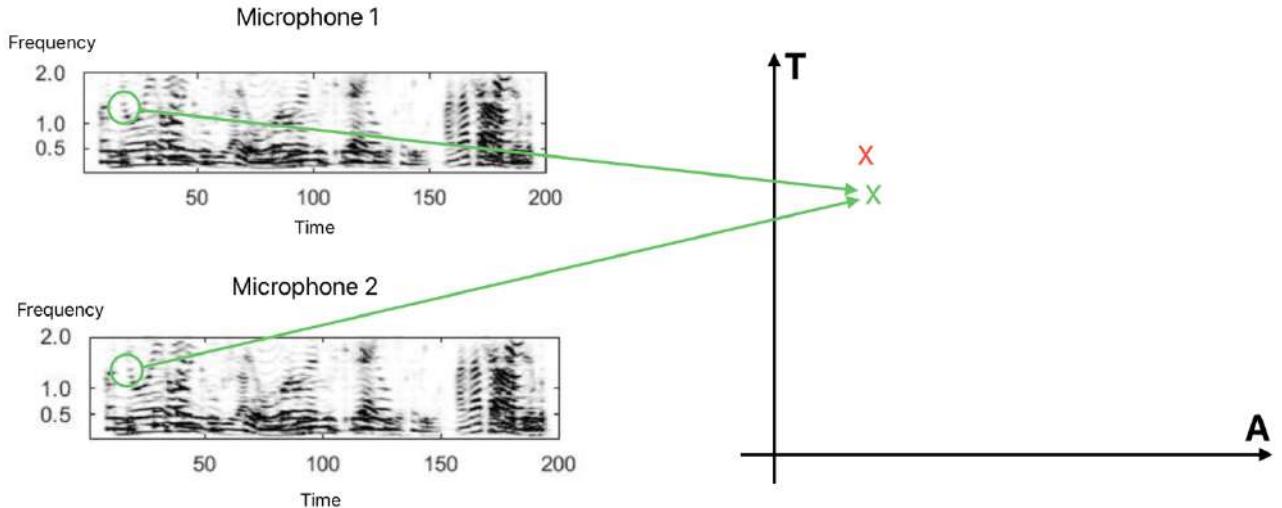
Let us estimate these two features from every time-frequency bin. Simply speaking, in our STFT image, we estimate a value of attenuation ratio and time-delay for every pixel.

$$\text{Amplitude ratio: } A(n, \omega_k) = \frac{|Y_2(n, \omega_k)|}{|Y_1(n, \omega_k)|} \quad \text{Time delay: } T(n, \omega_k) = \frac{\arg(Y_2(n, \omega_k)) - \arg(Y_1(n, \omega_k))}{\omega_k}$$

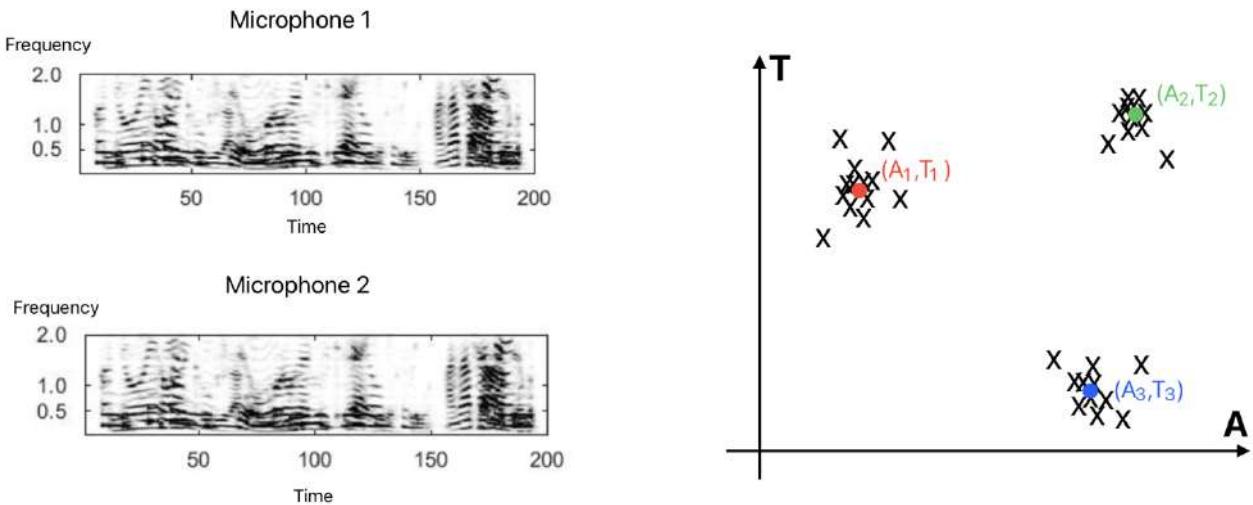
By considering the same time-frequency-bin on the STFT of the two microphones, we compute  $A(n, \omega_k)$  and  $T(n, \omega_k)$  that we represent as a point on a graph whose axis are the amplitude ration and time delay.



Then, we change the time-frequency bin and we do the same:



By iterating the algorithm, we notice that they cluster in 3 regions in the  $(A, T)$  plane, whose centroids are the pairs  $(A_i, T_i)$   $i = 1, 2, 3$ . Beware! This happens in this particular example; it is not written in stone that the regions in which the points cluster are three. This is a particular case from which abstract the general functioning.



At this point, we can use a clustering algorithm (e.g. k-means) for associating each entry to the source.

Reminder: k-means algorithm

Given a set of observations  $(\underline{x}_1, \underline{x}_2, \dots, \underline{x}_n)$ , where each observation is a  $d$ -dimensional real vector, k-means clustering aims to partition the  $n$  observations into  $k$  ( $\leq n$ ) sets  $\underline{S} = \{S_1, S_2, \dots, S_k\}$  so as to minimize the within-cluster sum of squares (WCSS) (i.e. variance). Formally, the objective is to find:

$$\arg \min_{\underline{S}} \sum_{i=1}^k \sum_{\underline{x} \in S_i} \|\underline{x} - \underline{\mu}_i\|^2 = \arg \min_{\underline{S}} \sum_{i=1}^k |S_i| \text{Var } S_i$$

Where  $\underline{\mu}_i$  is the centroid of points in  $S_i$  defined as

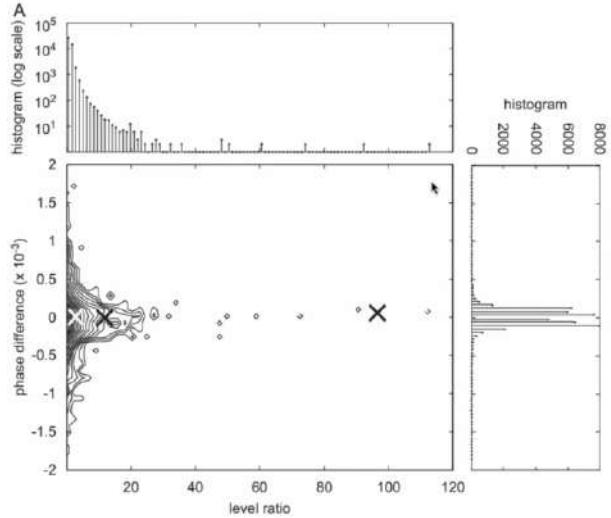
$$\underline{\mu}_i = \frac{1}{|S_i|} \sum_{\underline{x} \in S_i} \underline{x}$$

Where, in turns,  $|S_i|$  is the size of  $S_i$  and the operator  $\|\cdot\|$  is the usual  $L^2$  norm.

After that, we can compute the binary masks accordingly:

$$M_i(n, \omega_k) = \begin{cases} 1, & \text{if } [A(n, \omega_k), T(n, \omega_k)] \text{ belongs to cluster } [A_i, T_i] \\ 0, & \text{otherwise} \end{cases}$$

Actually, this does not work. On the right we report what we get. We can notice that the distribution of these features are very different from each others. This make very hard to separate them. We end up with this.



In order to solve this problem and maximise the ability of the cluster algorithm, we have to work synergically with the algorithm itself and the features. It is important to carefully choose the features in order to maximise the performance of the algorithm. For example, K-means is sensitive (basically it does not work) when one feature has a larger variance than the others. Why does not K-means work in this condition? K-means, in order to identify the position of the centroid makes some guesses. Then, it adapts the position according to data. If those data are on one axis “too close one to each other” and on the other one “too far”, the result is wrong since one coordinate “covers” the other. (This concept is best explained in the lecture related to the features).

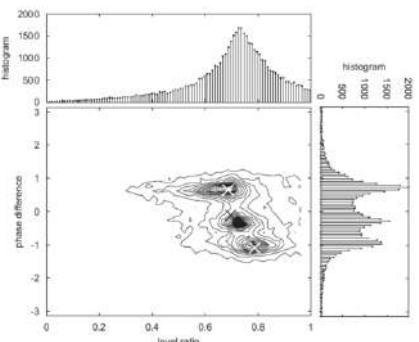
As we said, to solve this problem we can resort to different features. Quite often clustering may happen in a multi-dimensional space. This means that, in our case, we will not use just two single features (despite they are the most intuitive one) to represent the data, but we will assign each time-frequency bin with particular functions as:

$$\Phi(n, \omega_k) = \begin{bmatrix} A_1(n, \omega_k) \\ A_2(n, \omega_k) \\ P(n, \omega_k) \end{bmatrix} \in \mathbb{R}^3$$

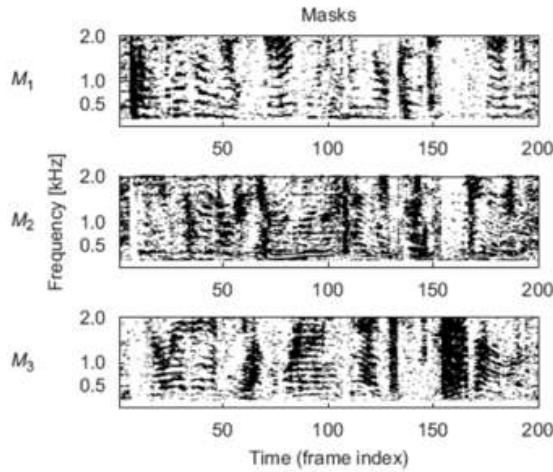
$$A_i(n, \omega_k) = \frac{|Y_i(n, \omega_k)|}{\sqrt{\sum_{m=1}^M |Y_m(n, \omega_k)|^2}}$$

$$P(n, \omega_k) = \frac{1}{2\pi} \arg\left(\frac{Y_2(n, \omega_k)}{Y_1(n, \omega_k)}\right)$$

We notice that these features are nothing but the previous ones but normalised. Since this makes them comparable, we get a better result as we can notice in the image on the right. The feature related to the amplitude is the normalised amplitude at the microphone (so it goes from zero to one) and then we have a normalised phase.

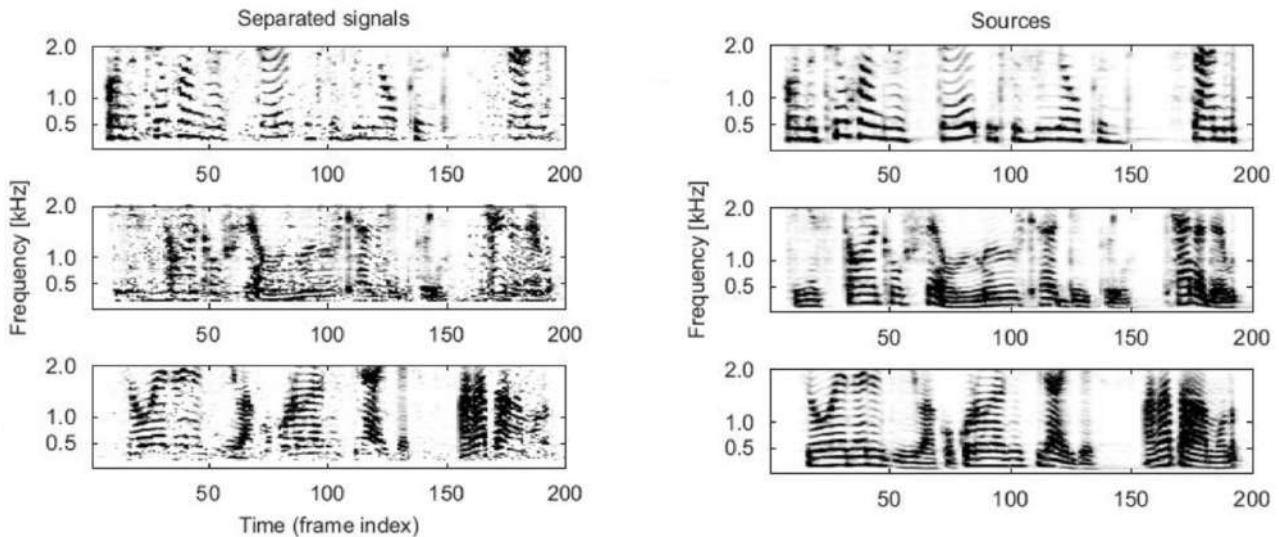


To resume what we learnt from this example we can say that, in general, the higher the dimension of the space, the easier becomes to separate data. Furthermore, it is better to normalise features.



On the left are reported the masks obtained with this technique. As we can see we have some points sparsely populated.

By applying the masks to the microphone signals, we can obtain an estimation of the sources.



Of course, the estimation is valid only for the magnitude since our binary mask has phase equal to zero (1 and 0 are both real positive number). Ignoring the phase evolution could affect the final result as the ear is able to hear the artefacts that it generates.

### Independent Component Analysis (ICA)

The next technique is a little bit complex but, at the same time, one of the most used. The basic assumption is the independence of the sources (as the name could suggest). Let us start by describing the scenario:

- Overestimated problem ( $M \geq N$ ): in our case  $M = N = 2$ .
- No information about the mixing matrix  $\rightarrow$  BSS
- Non reverberant environment  $\rightarrow$  instantaneous mix

The instantaneous mixing, in vectorial form, is represented by a multiplication between the signal and the mixing matrix:

$$\underline{y} = \underline{A} \underline{s}$$

Our goal is to obtain a de-mixing matrix  $\underline{\underline{W}} = \underline{\underline{A}}^{-1}$ , without any knowledge on source signals, such that:

$$\underline{s} = \underline{\underline{W}} \underline{y}$$

How can we estimate  $\underline{\underline{W}}$ ? Firstly, we need to make two hypotheses:

- Sources are independent (which implies that they are also uncorrelated) so the cross-correlation  $\underline{\underline{R}}_s = E\{\underline{s} \underline{s}^T\}$  is diagonal. Simply speaking, people during in dialogue do not say same things at the same time.
- Sources have non-gaussian statistics. Simply speaking, again, people do not produce noise when they are speaking. What they are telling has a certain order.

When dealing with this problem we have to consider some difficulties that cannot be solved. They are the so-called ambiguities:

1. We cannot determine variances of independent components:  $\underline{s}$  and  $\underline{\underline{A}}$  are unknown: any scalar multiplier in one of the sources can be cancelled by dividing the corresponding column of  $\underline{\underline{W}}$  for the same quantity.
2. We cannot determine the order of independent components:  $\underline{s}$  and  $\underline{\underline{A}}$  are unknown: we can freely change the order of the terms in the sum  $\underline{y} = \underline{\underline{A}} \underline{s}$  without any influence on the final result  $\underline{y}$ . This is a problem in practical application!

What we can do is to assume that  $\underline{s}$  has unitary variance so that the actual volume of the source will be compensated in the mixing matrix.

The key concept is the one of independence. It means that the joint-probability of the function is equal to the product of the marginals: knowing  $Y_1$  we have no information on  $Y_2$  and viceversa. If they are independent are also uncorrelated. Formally

$$p(y_1, y_2) \underset{\text{Indipendence}}{\equiv} p_1(y_1)p_2(y_2) \Rightarrow C_{Y_1 Y_2} = R_{Y_1 Y_2} - \eta_{Y_1} \eta_{Y_2} = E\{y_1 y_2\} - E\{y_1\}E\{y_2\} = 0$$

If the sources are independent one from each other, we can think at the mixture as a linear combination of the signals coming from them. Let us dwell on this concept with an example. We assume that  $S_1$  and  $S_2$  are two random variables that follows an uniform distribution.

Reminder: uniform distribution

Definition:

$$X \in \mathcal{U}[a, b]$$

Probability density function:

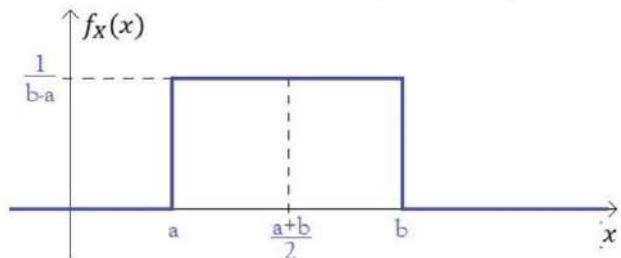
$$f_X(x) = \frac{1}{b-a} \text{rect}\left(\frac{x-\eta_X}{b-a}\right)$$

Mean value:

$$\eta_X = \frac{a+b}{2}$$

Variance:

$$\sigma_X^2 = \frac{(b-a)^2}{12}$$

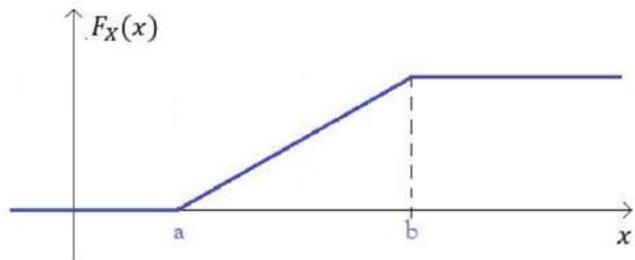


Statistical power:

$$p_x = \frac{a^2 + ab + b^2}{3}$$

Simply speaking, a random variable that follows an uniform distribution can assume every value in the interval  $[a, b]$  with the same probability.

Probability distribution function:



In this condition, if we know  $s_1$  we cannot say anything about  $s_2$ . This is evident just by looking at the graphical representation of the two variables on a same plane: by fixing one value on the  $s_1$  axis we notice that the variable  $s_2$  can always assume every value within its interval of definition independently on the value fixed on  $s_1$ . This means that  $s_1$  has no “influence” on  $s_2$ . However, if we mix the tracks (that mathematically correspond to multiply a mixing matrix to our source vector getting an algebraical transformation that corresponds to a rotation and a stretch in the vectorial space whose bases are  $y_1$  and  $y_2$ ) the two variables become not more independent. This time again it is evident by fixing a point on the  $y_1$  axis: the interval in which  $y_2$  can vary depends on the value fixed on  $y_1$ .

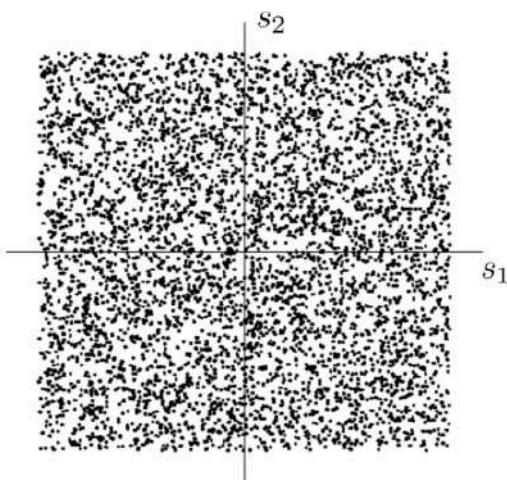
Joint distribution of two independent components that are uniformly distributed according to the p.d.f.

$$f_{s_1}(s_1) = \frac{1}{2\sqrt{3}} \text{rect}\left(\frac{s_1}{2\sqrt{3}}\right) \quad f_{s_2}(s_2) = \frac{1}{2\sqrt{3}} \text{rect}\left(\frac{s_2}{2\sqrt{3}}\right)$$

Joint distribution of the two mixtures, which are no more independent.

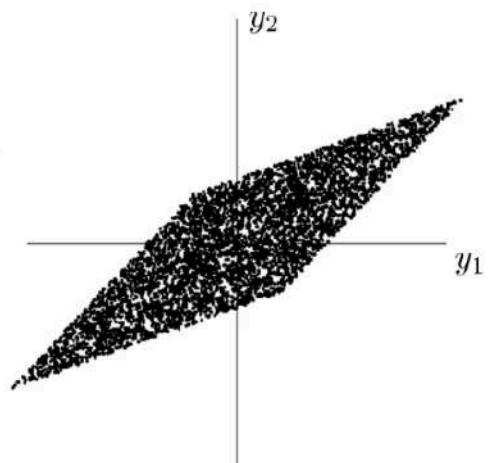
That means that:

$$f_{s_1 s_2}(s_1, s_2) = \begin{cases} \frac{1}{2\sqrt{3}}, & \text{if } s_1 \in [-\sqrt{3}, \sqrt{3}] \text{ and } s_2 \in [-\sqrt{3}, \sqrt{3}] \\ 0, & \text{otherwise} \end{cases}$$



Mixing process:

$$\underline{y} = \underline{A} \underline{s} \\ \underline{A} = \begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix}$$



We remind again that this result is a particular one: it is an example! The shape we get at the end depends on the distribution we have at the beginning and on the mixing matrix used. In light of this we can ask ourselves whether this process works with every distribution. The answer is no. If the two variables are Gaussian, the process fails. But why? Because they will be unaffected by any mixing matrix: after a rotation they remain as they are.

Reminder: Gaussian distribution

Definition:

$$X \in \mathcal{N}[\eta, \sigma^2]$$

Mean value:

$$\eta_X = \eta$$

Variance:

$$\sigma_X^2 = \sigma^2$$

Statistical power:

$$p_X = \sigma^2 + \eta$$

Simply speaking, a random variable that follows a Gaussian distribution has higher probability to assume a value closer to its mean value than to assume other values. The probability decreases the further we are from the mean value.

Transformation of Gaussian variables:

Given a vector of joint Gaussian variables  $(X_1, X_2)$ , by applying a linear transformation

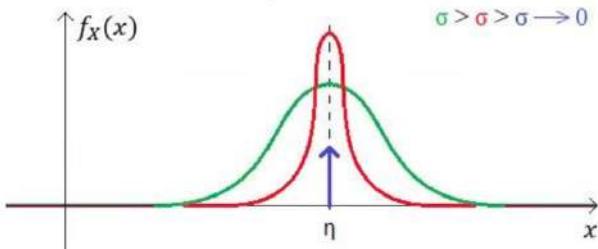
$$\underline{G} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \text{ we get a vector of joint Gaussian variables } (Y_1, Y_2).$$

Formally,  $(X_1, X_2) \in \mathcal{N}(\eta_{X_1}, \eta_{X_2}, \sigma_{X_1}^2, \sigma_{X_2}^2, \rho_{X_1 X_2})$

$$\begin{aligned} Y_1 &= a_{11}X_1 + a_{12}X_2 & \Rightarrow & \quad Y_1 \text{ Gaussian} \\ Y_2 &= a_{21}X_1 + a_{22}X_2 & \Rightarrow & \quad Y_2 \text{ Gaussian} \end{aligned}$$

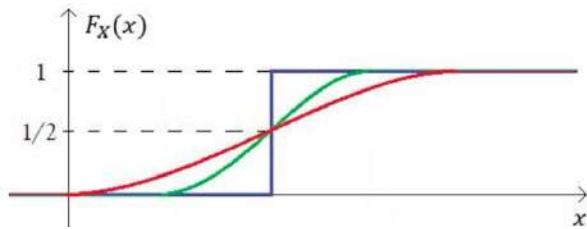
Probability density function:

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\eta)^2}{2\sigma^2}}$$



Probability distribution function:

$$F_X(x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\eta)^2}{2\sigma^2}} dx$$



By considering a variable of this kind, what it happens is what it follows.

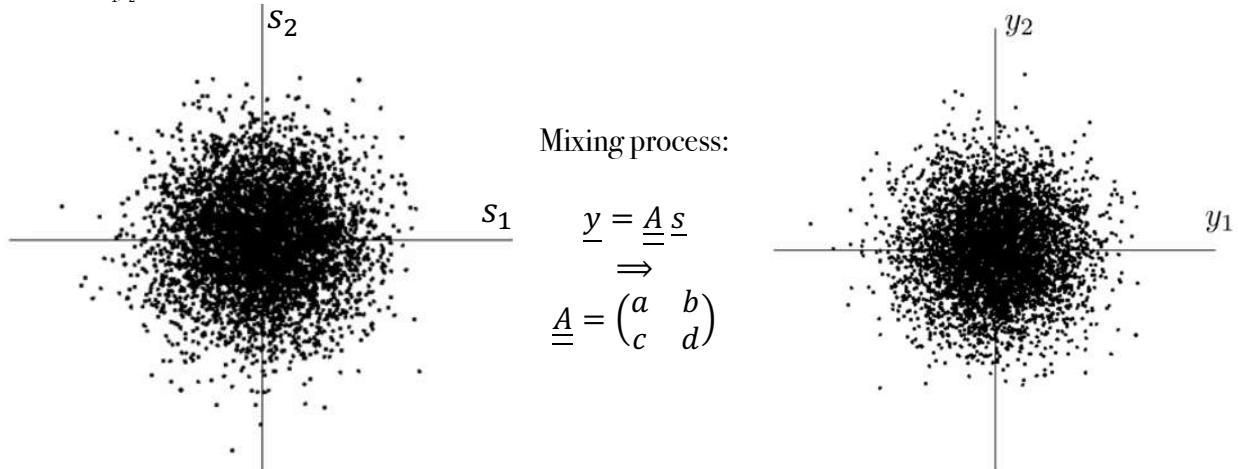
Joint distribution of two independent ( $\rho_{S_1, S_2} = 0$ ) components that are uniformly distributed according to the p.d.f.

$$f_{S_1}(s_1) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(s_1-\eta)^2}{2\sigma^2}} \quad f_{S_2}(s_2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(s_2-\eta)^2}{2\sigma^2}}$$

Joint distribution of the two mixtures, which are no more independent.

That means that:

$$\underset{\rho_{S_1, S_2}=0}{f_{S_1, S_2}(s_1, s_2)} = f_{S_1}(s_1)f_{S_2}(s_2) = \frac{1}{2\pi\sigma^2} e^{-\frac{(s_1-\eta)^2+(s_2-\eta)^2}{2\sigma^2}}$$



Where  $\underline{A}$  is an orthogonal matrix.

As we can notice, the joint density is now symmetric with respect to the origin, thus it does not give any information about the direction of the columns of the mixing matrix. If the source variables are Gaussian, any orthogonal mixing matrix leads to mixed component which are independent and Gaussian.

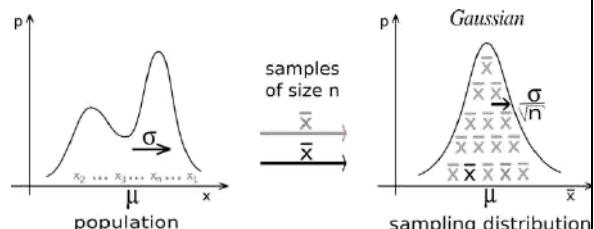
Separation is not possible in this case. And this is a problem! But why? Are all the signals Gaussian? In general they are not, but when we consider the sum of independent random variables what we get is a Gaussian distribution according to the Central Limit Theorem.

**Reminder: central limit theorem**

The central limit theorem (CLT) states that, under appropriate conditions, the distribution of a normalized version of the sample mean converges to a standard normal distribution. This holds even if the original variables themselves are not normally distributed.

This theorem can find an application on the sound of rain.

Each raindrop that hit a surface generates its own peculiar sound, independently from the other. The overall result from the sum of all the sounds, resemble white noise (described by a Gaussian model).



So, when we have a sum of signals, it is highly probable that it tends towards a Gaussian. To solve this problem, we rewrite the mixture of the microphone signals as

$$x = \underline{w}^T \underline{y} = \underline{w}^T \underline{A} \underline{s}$$

where  $x$  is surely more Gaussian than  $\underline{s}$  accordingly to what we have just said. Surely, it cannot be lower than the one of  $\underline{s}$  as it contains the signal that we mix together to obtain  $x$ . In this expression, we decide to introduce this new vector  $\underline{w}$  in order to maximise the non-Gaussianity of  $x$ . Ideally, perfect separation is

achieved when  $w$  correspond to one of the rows of the de-mixing matrix  $\underline{\underline{W}} = \underline{\underline{A}}^{-1}$  (that in reality we do not know since we are working in BSS conditions).

It is really nice to have an idea on how the vector  $\underline{w}$  should be as it can allows us to understand how good the result is. So, everything boils down to how to measure gaussianity that means finding a measure that tells us how much  $x$  follows a Gaussian distribution.

### Kurtosis

In order to do so, we introduce the Kurtosis (from Greek: κυρτός, meaning "curved, arching") that gives us a measure of the "tailedness" of the probability distribution of a real-valued random variable. Reminding that the  $k$ -th normalised central statistical moment is defined as

$$E \left\{ \left( \frac{X - \eta_X}{\sigma} \right)^k \right\}$$

we define the kurtosis as the fourth standardised moment:

$$\text{Kurt}\{X\} \triangleq E \left\{ \left( \frac{X - \eta_X}{\sigma_X} \right)^4 \right\} = E \left\{ \frac{(X - \eta_X)^4}{\sigma_X^4} \right\} \underset{\sigma_X^2 \triangleq E\{L_X^2\} = E\{(X - \eta_X)^2\}}{=} \frac{E\{(X - \eta_X)^4\}}{(E\{(X - \eta_X)^2\})^2} \underset{\mu_k = E\{(X - \eta_X)^k\}}{=} \frac{\mu_4}{\sigma^4} \underset{\substack{\text{Central} \\ \text{Moment}}}{=}$$

In our case ( $\eta_X = 0$ ):

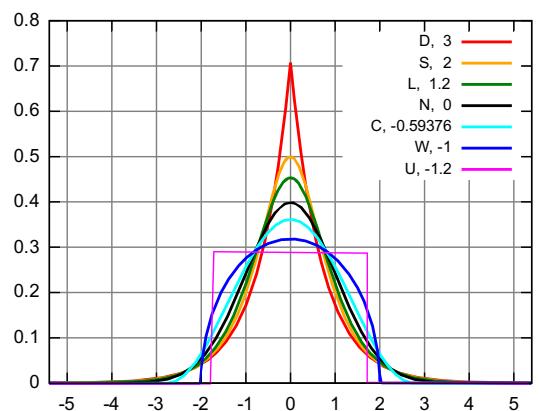
$$\text{Kurt}\{X\} = E\{X^4\} - 3(E\{X^2\})^2$$

If:

- $\text{Kurt}\{X\} < 0$ ,  $X$  is sub-Gaussian whose extreme case is the uniform distribution  $\mathcal{U}$ .
- $\text{Kurt}\{X\} = 0$ ,  $X$  follows a Gaussian distribution;
- $\text{Kurt}\{X\} > 0$ ,  $X$  is super-Gaussian that is a very peaky distribution like the Laplace's one  $\mathcal{D}$ .

In the graph on the right, we can see how Kurtosis is worth for different types of distribution that are:

- $\mathcal{D}$ : Laplace Distribution
- $\mathcal{S}$ : Hyperbolic Secant Distribution
- $\mathcal{L}$ : Logistic Distribution
- $\mathcal{N}$ : Normal Distribution
- $\mathcal{C}$ : Raised Cosine Distribution
- $\mathcal{W}$ : Wigner Semicircle Distribution
- $\mathcal{U}$ : Uniform Distribution



In our case, we can take its absolute value as a non-Gaussianity index: the higher  $|\text{Kurt}\{X\}|$ , the better.

### *Negentropy*

Another metric that we can check is Negentropy. We know that a Gaussian Variable has the largest entropy among all random variables of equal variance. Why that? We know that the entropy tells us how much information is provided by a certain source. Since the Gaussian distribution represents a random process, we cannot retrieve any information on the next sample of a signal by looking at the previous samples. For this reason, all the information will come from the source maximising the entropy.

So, the entropy of a random variable  $X$ , given a pdf  $f_X(x)$ , is equal to:

$$H(X) = - \int_{-\infty}^{\infty} f_X(x) \log f_X(x) dx$$

Given a Gaussian random variable  $X_{Gaussian}$  with the same variance of  $X$ , the negentropy is defined as

$$J(x) = H(X_{Gaussian}) - H(X)$$

That can be approximated in function of the Kurtosis value of  $X$

$$J(X) \approx \frac{1}{12} E\{X^2\}^2 + \frac{1}{48} \text{Kurt}\{X\}^2$$

Our aim is to maximise the Nagentropy in order to maximise the Non-Gaussianity of  $X$ .

### *Mutual Information*

Another measure that we can take into account to understand how much Gaussian a variable is, is the mutual information that measures the dependence between random variables

$$I(X_1, X_2, \dots, X_N) = \sum_{i=1}^N H(X_i) - H(\underline{X})$$

The mutual information is always non-negative, and zero if variables are statistically independent. For what it concerns Gaussian distribution, we can affirm that should be minimised since it is maximal if it follows the Gaussian distribution. The sum of the entropies has to be smaller than the entropy given by the whole series. Based on what we said, we know that the entropy of these variables should be maximised, which corresponds to the energy of each single variable if they followed the Gaussian distribution. So, knowing any of those variables doesn't tell us any information about another realisation. The sum of Ola is the singular. The entropy of the single variables should be always smaller than this. Entropy given by the whole of serious series because we assumed that actually we don't have a real Gaussian distribution. So these should be always 1 negativa and it will be 0 if all the variables in the vector  $X$  are statistically independent, which means that they belongs to that distribution and these will be basically these  $X$ . The entropy of the whole Vectra will be word. The sum of the entropy of its single variable.

### Kullback-Leibler divergence

This is the most natural measure of independency as it measures it as the distance between the joint vectorial probability density function and the product of the marginal densities. To understand what it means, let us see it in two dimensions. The joint p.d.f. of two variables is defined as

$$f_{X_1, X_2}(x_1, x_2) \triangleq \frac{\partial^2 F_{X_1, X_2}(x_1, x_2)}{\partial x_1 \partial x_2}$$

Where:

$$F_{X_1, X_2}(x_1, x_2) \triangleq P[X_1 \leq x_1, X_2 \leq x_2]$$

The only case in which the product of the marginals is equal to the joint probability function is when we are dealing with a Gaussian joint pdf and the two marginals are independent ( $\rho_{X_1, X_2} = 0$ ). This is the only case in which  $f_{X_1, X_2}(x_1, x_2) = f_{X_1}(x_1)f_{X_2}(x_2)$ . So, in this case the distance between the function  $f_{X_1, X_2}(x_1, x_2)$  and the product  $f_{X_1}(x_1)f_{X_2}(x_2)$  will be zero. Since we want to avoid Gaussian variables, this divergence has to be as highest as possible. Formally, for distributions  $F_X(x)$  and  $F_Y(y)$  of a continuous random variable, the Kullback-Leibler divergence is defined as

$$D_{KL}(F_X(x) \parallel F_Y(y)) = \int_{-\infty}^{\infty} f_X(x) \log \left( \frac{f_X(x)}{f_Y(x)} \right) dx$$

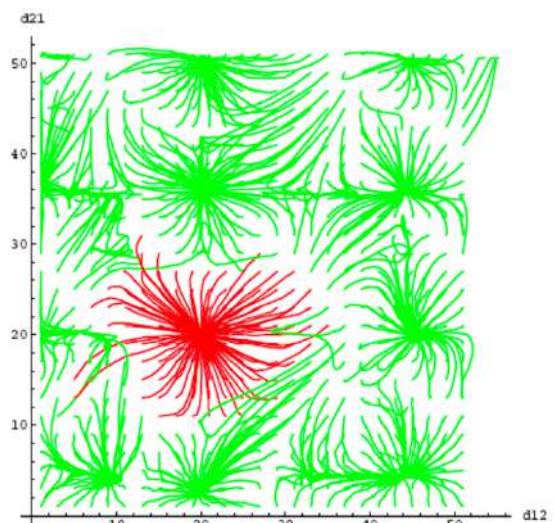
### Estimation of the De-mixing matrix

Once we have determined a suitable cost function (kurtosis, negentropy, mutual information, ...), the estimation of the demixing model is achieved by solving an optimization problem:

$$\hat{\underline{w}} = \arg \min_{\underline{w}} J(\underline{w}) \quad \text{or} \quad \hat{\underline{w}} = \arg \max_{\underline{w}} J(\underline{w})$$

Depending on the chose independence metric.

This cost function are typically not represented by bell-shaped functions. the cost function is strongly not-linear, therefore the iterative algorithm could be trapped into a local minimum/maximum. This means that we are not able to find a close form solution. In fact, the demixing model is typically estimated through LMS-like iterative algorithms. The solution, therefore, may depend on the initial conditions adopted in each realisation. On the right are reported some example of convergence starting from different initial conditions. As we can notice, the point in which the lines converge are not only one. After having explored all the possible minima, we compare them and we chose the smallest. In this example, the red one.



### References

- Shoko Araki et.al., “Undetermined blind sparse source separation for arbitrarily Arranged multiple sensors”
- Hyvarinen A., “A survey on Indipendent Component Analysis ”, Neural Computing Surveys, 2: 94-128
- Hyvarinen A., “Independent Component Analysis: Algorithmsand Applications ”, available at:  
[http://www.cis.hut.fi/aapo/papers/IJCNN99\\_tutorialweb/](http://www.cis.hut.fi/aapo/papers/IJCNN99_tutorialweb/)

## Exercises

19<sup>th</sup> March 2024  
Mirco Pezzoli

### Exercise on Wiener Filtering

#### First Exercise

We want to implement an acoustic echo cancellation system based on the Wiener filter. The microphone signal is described as follows:

$$u(n) = 0.4v(n) + 0.15v(n-1) + b(n)$$

where  $b(n)$  represents the speech signal and  $v(n)$  is the echo component emitted by the loudspeaker. For simplicity, we assume that  $b(n)$  and  $v(n)$  are mutually uncorrelated within the observation window. The autocorrelation function of  $v(n)$  has been estimated as  $r(0) = 0.8$ ,  $r(1) = 0.36$ ,  $r(k) = 0$  for  $k = 2$ . The echo canceler has to estimate the speech signal by means of a Wiener filter with 2 samples.

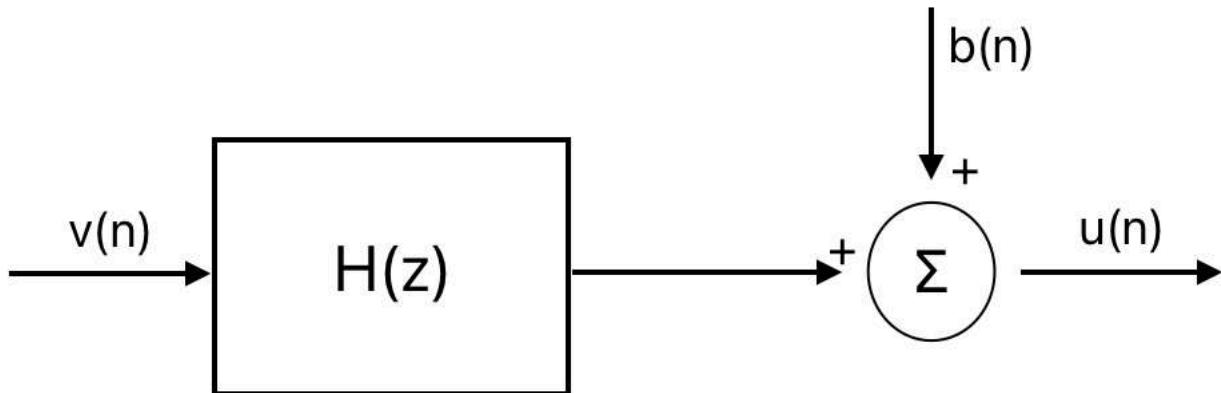
#### Questions:

- a) Compute the autocorrelation matrix  $\mathbf{R}$  of the input signal, and the cross-correlation vector  $\mathbf{p}$  of the input signal and the desired response.
- b) Compute the set of the optimum (in the MSE sense) filter coefficients, namely  $w(i)$  with  $i = 1, 2$ .

#### Solution:

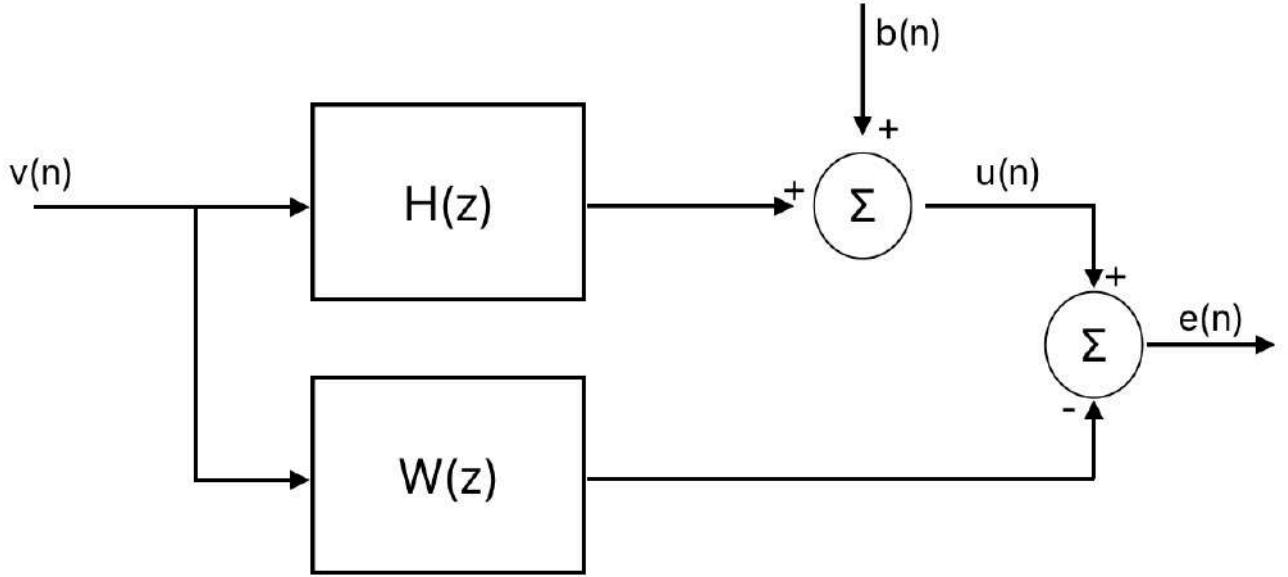
We start by saying that  $b(n)$  is what we can capture with the microphone.

What we have here is that  $u(n) = 0.4v(n) + 0.15v(n-1) + b(n)$ . So firstly, we write the block scheme.



We assume that the filter  $H$  is unknown.

We need to find the Wiener Filter to cancel the noise. By considering  $v(n)$  as our reference signal, the overall scheme (considering also the Wiener filter) will be



Now the point is that, we have other information:

- $b(n) \perp v(n)$
- $r_v(0) = 0.8; r_v(1) = 0.36; r_v(k) = 0 \forall k > 2$

The desired response is  $u(n)$ .

We need to write the autocorrelation matrix. The autocorrelation matrix:

- Has entrees that corresponds to the autocorrelation
- is symmetric

$$\underline{R} = \begin{pmatrix} r(0) & r(1) \\ r(1) & r(0) \end{pmatrix} = \begin{pmatrix} 0.8 & 0.36 \\ 0.36 & 0.8 \end{pmatrix}$$

We now have to find the cross correlation

$$\begin{aligned}
 p(k) &= E\{v(n)u(n-k)\} \\
 &= E\{v(n)[0.4v(n-k) + 0.15v(n-1-k) + b(n-k)]\} \\
 &= 0.4\underbrace{E\{v(n)v(n-k)\}}_{r(k)} + 0.15\underbrace{E\{v(n)v(n-1-k)\}}_{r(k+1)} + \underbrace{E\{v(n)b(n-k)\}}_{=0} \\
 &= 0.4r(k) + 0.15r(k+1)
 \end{aligned}$$

In vectorial fashion

$$\underline{p} = \begin{pmatrix} p(0) \\ p(-1) \end{pmatrix} = \begin{pmatrix} 0.4r(0) + 0.15r(1) \\ 0.4r(1) + 0.15r(0) \end{pmatrix} = \begin{pmatrix} 0.347 \\ 0.264 \end{pmatrix}$$

We shift cross correlation in  $-1$  for convention but, since it is symmetric, it would be the same if we used  $p(1)$  instead of  $p(-1)$ .

Now we have all we need to find the coefficients. We just have to invert the relation below

$$\underline{w} = \underline{\underline{R}}^{-1} \underline{p}$$

Reminder: how to find the inverse of a Matrix

Given a matrix  $A$ , the inverse of  $A$  is equal to

$$A^{-1} = \frac{\text{adj } A}{\det A}$$

Where  $\text{adj } A$  is the adjoint matrix of  $A$ , so the transpose of its cofactor matrix where the cofactor matrix is, in turn, the matrix containing the cofactors of each of the elements of the given matrix.

Simply speaking, to inverse a matrix  $2 \times 2$   $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$  we follow this path:

- 1) We assure that the matrix exists, this happens if  $ad - bc = 0$ ,
- 2) We apply the formula

$$A^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

If the matrix has a greater dimension, we apply the Gauss-Jordan algorithm: we put aside to the matrix we want to invert the identity matrix and we works in Gauss' way in order to make the identity matrix shift on the left. We report an example to better understand the process.

$A$  is the matrix we want to invert

$$A = \begin{pmatrix} 1 & 2 & 0 \\ 1 & -1 & 1 \\ 2 & 1 & 0 \end{pmatrix}$$

We put it aside to the identity matrix

$$\left( \begin{array}{ccc|ccc} 1 & 2 & 0 & 1 & 0 & 0 \\ 1 & -1 & 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 & 0 & 1 \end{array} \right)$$

We work in Gauss' way in order to shift the matrix

$$\begin{aligned} A|I &= \left( \begin{array}{ccc|ccc} 1 & 2 & 0 & 1 & 0 & 0 \\ 1 & -1 & 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 & 0 & 1 \end{array} \right) \xrightarrow[R_3 - 2R_1]{R_1 - R_2} \left( \begin{array}{ccc|ccc} 1 & 2 & 0 & 1 & 0 & 0 \\ 0 & 3 & -1 & 1 & -1 & 0 \\ 0 & -3 & 0 & -2 & 0 & 1 \end{array} \right) \rightarrow \\ &\xrightarrow[R_3 + R_2]{R_3 + R_2} \left( \begin{array}{ccc|ccc} 1 & 2 & 0 & 1 & 0 & 0 \\ 0 & 3 & -1 & 1 & -1 & 0 \\ 0 & 0 & -1 & -1 & -1 & 1 \end{array} \right) \xrightarrow[R_2 - R_3]{R_2 - R_3} \left( \begin{array}{ccc|ccc} 1 & 2 & 0 & 1 & 0 & 0 \\ 0 & 3 & 0 & 2 & 0 & -1 \\ 0 & 0 & -1 & -1 & -1 & 1 \end{array} \right) \rightarrow \\ &\xrightarrow[3R_3 + 2R_2]{3R_3 + 2R_2} \left( \begin{array}{ccc|ccc} 3 & 0 & 0 & -1 & 0 & 2 \\ 0 & 3 & 0 & 2 & 0 & -1 \\ 0 & 0 & -1 & -1 & -1 & 1 \end{array} \right) \xrightarrow[-R_3]{\frac{1}{3}R_1, \frac{1}{3}R_2} \left( \begin{array}{ccc|ccc} 1 & 0 & 0 & -\frac{1}{3} & 0 & \frac{2}{3} \\ 0 & 1 & 0 & \frac{2}{3} & 0 & -\frac{1}{3} \\ 0 & 0 & 1 & 1 & 1 & -1 \end{array} \right) = I|A^{-1} \end{aligned}$$

To run a check, we just have to make the product between  $A$  and  $A^{-1}$  and see if returns  $I$

So

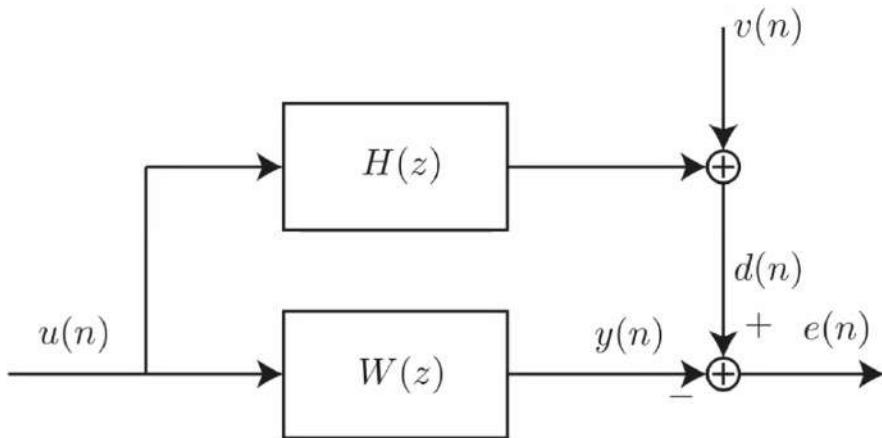
$$\underline{w} = \frac{1}{0.8^2 - 0.36^2} \begin{pmatrix} 0.8 & 0.36 \\ 0.36 & 0.8 \end{pmatrix} \begin{pmatrix} 0.347 \\ 0.264 \end{pmatrix} = \begin{pmatrix} 0.4 \\ 0.15 \end{pmatrix}$$

At this point we can compute the optimum MSE

$$\begin{aligned} e(n) &= u(n) - y(n) \\ &\stackrel{y(n)=\underline{w}(n)*v(n)}{=} \\ &= 0.4v(n) + 0.15v(n-1) + b(n) - 0.4v(n) - 0.15v(n-1) = b(n) \end{aligned}$$

### Second Exercise

Consider the system below, we know that  $H(z) = 2 + 3z^{-1}$ ,  $u \sim \mathcal{N}(0, \sigma_u^2 = 1)$ ,  $v \sim \mathcal{N}(0, \sigma_v^2 = 0.1)$ ,  $u \perp tv$  and  $u, v \in \mathbb{R}$



*Questions:*

- a) Calculate the optimum filter  $W_o(z)$  with  $M = 2$  taps.

*Solution:*

We need basically two things:

- Autocorrelation of  $u(n)$
- Cross correlation between  $u(n)$  and the desired signal

Firstly, we have to find which is the desired signal

$$\begin{aligned} v(n) &= d(n) - y(n) = u(n) * h(n) - u(n) * w(n) = \\ &= 2u(n) + 3u(n-1) + v(n) - u(n) * w(n) \end{aligned}$$

Then we evaluate the autocorrelation

$$r_u(k) = E\{u(n)u(n-k)\} \stackrel{\substack{\text{Being } u \\ \text{white-noise}}}{=} \begin{cases} 1, & k = 0 \\ 0, & \text{otherwise} \end{cases}$$

Hence

$$\underline{\underline{R}} = \begin{pmatrix} r(0) & r(1) \\ r(1) & r(0) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Let us now calculate the cross-correlation vector

$$\underline{p} = \begin{pmatrix} p(0) \\ p(-1) \end{pmatrix}$$

With

$$\begin{aligned} p(0) &= E\{u(n)d(n)\} = E\{u(n)[2u(n) + 3u(n-1) + v(n)]\} = \\ &= E\{2u^2(n)\} + \underbrace{E\{3u(n)u(n-1)\}}_{r(k=-1)=0} + \underbrace{E\{u(n)v(n)\}}_{\substack{=0 \\ u \perp tv}} = 2\sigma_u^2 + 0 + 0 = 2 \end{aligned}$$

And

$$\begin{aligned} p(-1) &= E\{u(n-1)d(n)\} = E\{u(n-1)[2u(n) + 3u(n-1) + v(n)]\} = \\ &= \underbrace{E\{2u(n)u(n-1)\}}_{r(k=-1)=0} + E\{3u^2(n-1)\} + \underbrace{E\{u(n-1)v(n)\}}_{\substack{=0 \\ u \perp tv}} = 0 + 3\sigma_u^2 + 0 = 3 \end{aligned}$$

Having both the autocorrelation matrix and the cross-correlation vector, we can compute the optimum filter coefficients as follows

$$\underline{w}_o = \underline{\underline{R}}^{-1} \underline{p} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 2 \\ 3 \end{pmatrix} \underset{I = I^{-1}}{\equiv} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}$$

Check: they are exactly equal to the ones of H.

**Exercises related to windowing***True or False*

- Consider the problem of resolving sinusoidal peaks in the frequency domain. Given a window with factor  $L = 2$  and two peaks at  $f_1 = 100 \text{ Hz}$  and  $f_2 = 300 \text{ Hz}$ . We need at least 160 samples to have  $F_s = 8 \text{ KHz}$ .

True or false?

$$M \geq 160$$

We know that  $M \geq L \frac{F_s}{f_2 - f_1} = 2 \frac{8000}{200} = 80$  so it is false. We are happy just with 80 samples.

- Slightly different flavour of the same question. Consider a harmonic signal with a fundamental frequency  $f_0 = 100 \text{ Hz}$ . It is acquired at 50 KHz. To resolve the spectral peaks with a rectangular window. At least 2000 samples are needed.

$$F_s = 50 \text{ KHz}$$

$L = 2$  rectangular window

$$M \geq 2000$$

We solve it using the same formula as before. We remind that in an harmonic signal, the distance between two peaks is equal to the fundamental frequency,

$$M \geq L \frac{F_s}{f_0} = 2 \frac{50000}{100} = 1000$$

We need at least 1000 samples. So, it is false again.

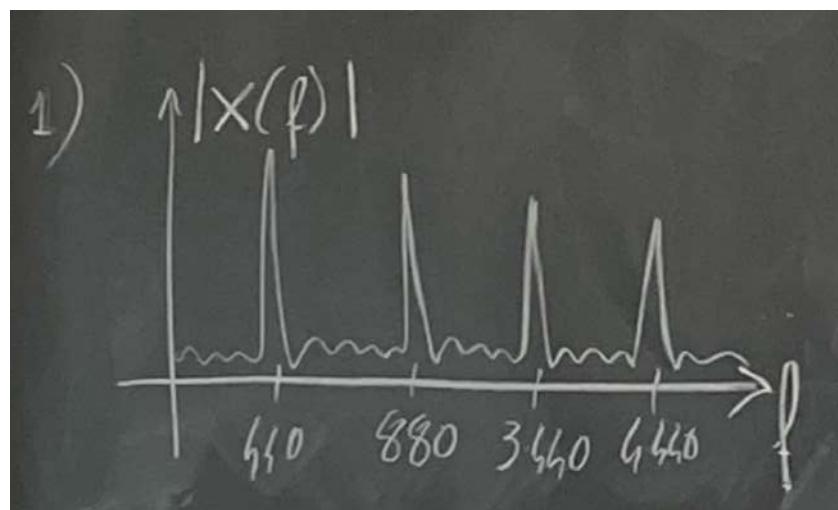
*Exercise*

You are developing a system for music transcription that works by tracking sinusoidal peaks in frequency domain. You are tuning your system to work with a specific instrument that reproduces harmonic sounds with the lowest possible fundamental frequency equal to 440 Hz.

- Sketch an approximate version of the spectrum obtained when playing the lowest pitch note.

$$f_{0\text{low}} = 440 \text{ Hz (A4)}$$

Since the signal that we are generating are harmonics, the spectrum will be a series of peaks at the harmonics. The number of harmonics depends on the instrument, the sampling frequency and how we low pass the signal. Nothing is specified.



- 2) Consider a rectangular window with  $L = 2$  and a sampling frequency of  $44 \text{ KHz}$ . Which is the length in seconds and in samples of the analysis window that allows you to resolve the sinusoidal peaks of the note A<sub>4</sub> note? Which is the condition related to the window main lobes to be satisfied in this situation?

$$L = 2$$

$$F_s = 44 \text{ KHz}$$

$$M_{\text{samp}} = ?$$

$$M_{\text{seconds}} = ?$$

$$f_0(\text{A}4) = 440 \text{ Hz}$$

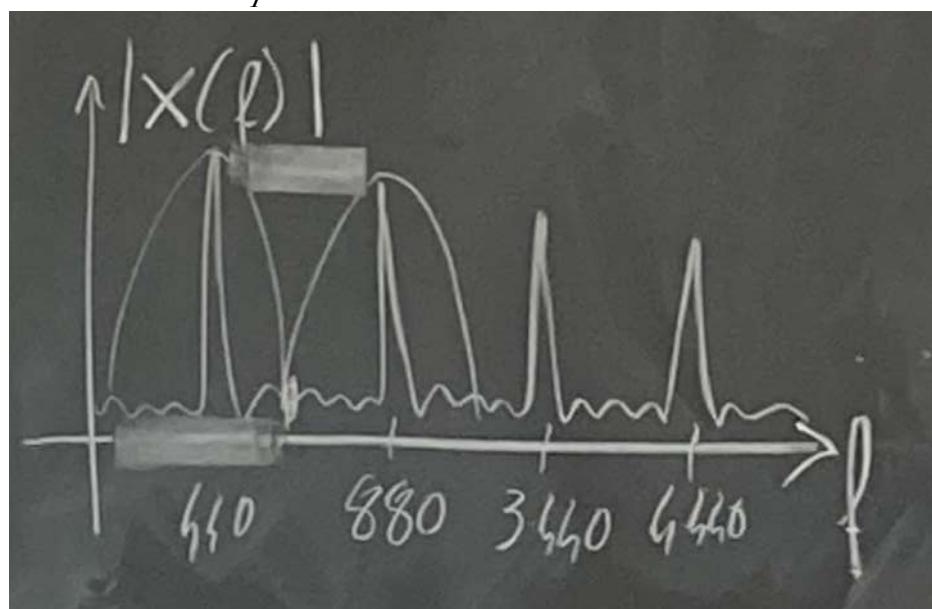
Same formula:

$$M_{\text{samp}} = 2 \cdot \frac{44000}{440} = 200$$

How many seconds? We divide  $M_{\text{samp}}$  by the sampling frequency.

$$M_{\text{seconds}} = \frac{200}{44000} = 4,5 \text{ ms}$$

The condition related to the lobe is that the windowing lobe width should at most equal to the distance between the two closest peaks.



- 3) How would you change the analysis window if the instrument lowest fundamental frequency drops at  $220 \text{ Hz}$ ?

$$f_{0\text{low}}^{\text{new}} = 220 \text{ Hz}$$

I simply have to multiply the samples by the same factor of which I had divided the frequency.

$$M_{\text{samp}}^{\text{new}} = 400$$

The other parameters cannot be changed in the analysis window since I should resample the signal or change the type of window used.

*Exercise*

You want to analyse the sound of a saxophone using a STFT analysis scheme. Your goal is to localise and extract the spectral lines and track them over time. Knowing that the frequency of the lowest note of a tenor sax is approximately  $\frac{2}{3}$  of that of an alto sax, which of the two instruments will be more computational demanding for the application and why? In particular, if  $M$  is the length of the window used for the alto sax, how long will the same window need to be in the case of the tenor sax and why?

$$f_{tenor} = \frac{2}{3} f_{alto}$$

$$M_{alto} = M$$

We try to compute the  $M$ -values and let us see which is the higher

$$M_{tenor} = L \cdot \frac{F_s}{f_{tenor}}$$

$$M_{alto} = M = L \cdot \frac{F_s}{f_{alto}}$$

Since  $f_{tenor} = \frac{2}{3} f_{alto}$

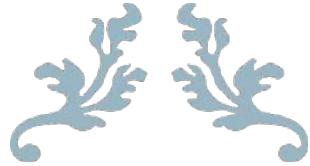
$$M_{tenor} = L \cdot \frac{F_s}{f_{tenor}} \rightarrow L \cdot F_s = M_{tenor} \cdot f_{tenor}$$

$$M_{alto} = M = L \cdot \frac{F_s}{f_{alto}} = M_{tenor} \cdot \frac{f_{tenor}}{f_{alto}} = \frac{2}{3} M_{tenor} \rightarrow M_{tenor} > M_{alto}$$

I will need more samples to work on the tenor sax. Therefore, the analysis of its sound will be more computational demanding.







---

# SOUND SYNTHESIS AND SPATIAL PROCESSING

---

Second Module



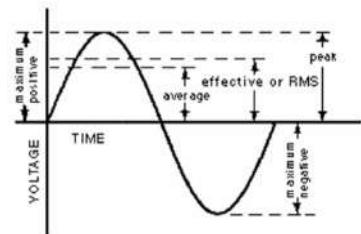
A.Y. 2023/2024  
POLITECNICO DI MILANO  
MUSIC AND ACOUSTIC ENGINEERING



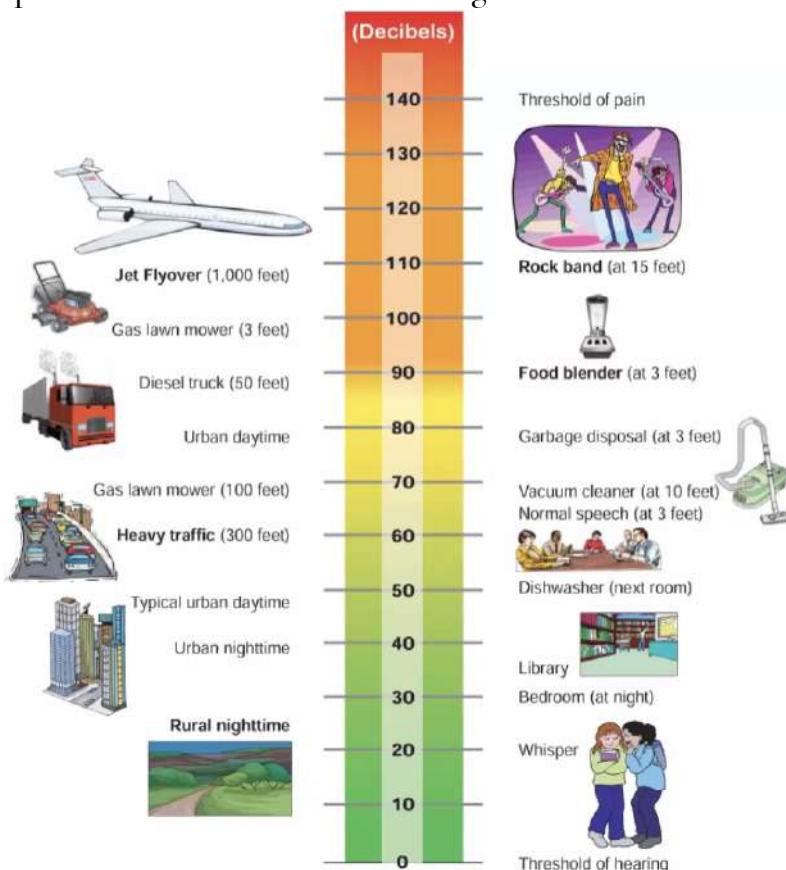
## Quick and (very) dirty recap on acoustics

It can be useful to introduce some values, for example the **root mean square value** that for a sinusoid is equal to

$$P_{eff} = \frac{P_0}{\sqrt{2}}$$



We report some examples to understand how various the range of SPL is.



In defining SPL level, it is important to consider that our perception in daytime and nighttime changes! The sound of a keyboard at daytime sounds like a mechanical hammer at nighttime.

KEYBOARD MODEL	Switch type	Peak sound level (dB)
Razer BlackWidow Ultimate Stealth Edition	Brown	59.3
Rosewill RK-9000	Blue	57.7
Adesso MKB-135B	Blue	57.3
Das Keyboard Model S Professional	Blue	57.0
Diatec Filco Majestouch 2 Camouflage	Blue	57.0
Diatec Filco Majestouch Ninja	Black	55.6
Thermaltake Meka/Meka G1	Black	54.6
Logitech membrane keyboard	n/a	54.2
SteelSeries 6Gv2/7G	Black	54.1
Microsoft Digital Media 3000 membrane keyboard	n/a	47.4

### Power of instruments in an orchestra

Another important parameter to be taken into account is the **energy per unit time** measured in Watt. These are for examples the powers of some instruments.

SOURCE	POWER (WATTS)	SOURCE	POWER (WATTS)
Orchestra (75 piece) @ fff	70	Orchestra (75 piece) @ mf	0.09
Bass Drum	25	Piccolo	0.08
Pipe Organ	13	Flute	0.06
Snare Drum	12	Clarinet	0.05
Cymbals	10	French Horn	0.05
Trombone	6	Triangle	0.05
Piano	0.4	Bass Voice	0.03
Bass Saxophone	0.3	Alto Voice (pp)	0.001
Bass Tuba	0.2	Average Speech [2]	0.000024
Double Bass	0.16	Violin (@ pppp)	0.0000038

### Intensity

Another important parameter is the intensity

$$I = \frac{P_{eff}^2}{\rho c}$$

Where:

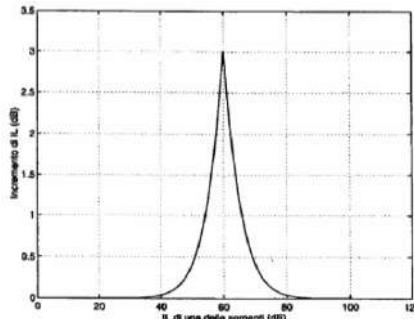
- $\rho$  is the density of the medium
- $c$  is the speed of sound.

For the intensity also, we can define the logarithmic representation.

$$IL = 10 \log \left( \frac{I}{I_{ref}} \right)$$

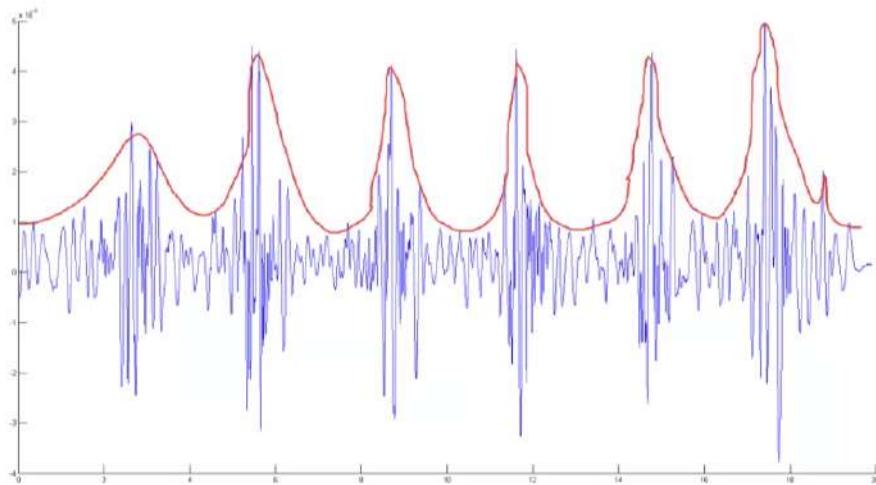
But what happens when we have more than one source?

Unfortunately, sourced does not interact linearly in terms of intensity. For example, given a source  $a$  with a  $IL = 60 \text{ dB}$  and a source  $b$  that goes from 0 to  $120 \text{ dB}$ , the increment on the resulting sound wave will be  $3 \text{ dB}$



## Envelope

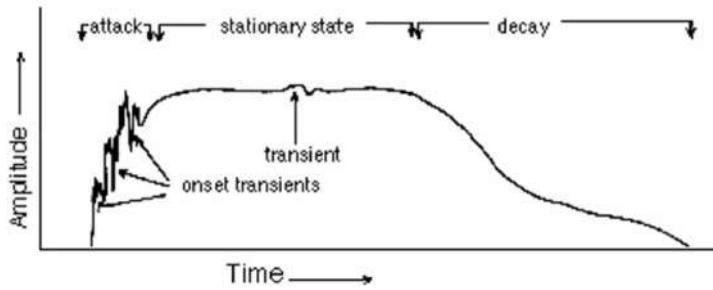
The envelope is nothing but the temporal evolution of the macro-level of the sound pressure (waveform).



In order to retrieve it, we use the Hilbert Transform.

If we consider the typical behaviour of a sound wave produced by a musical instrument, we can identify four main phases:

- Attack: from the beginning to the instant in which envelope reaches its maximum
- Transients: initial fluctuations
- Stationary state: also called sustain, from the end of the attack to the beginning of the decay
- Decay: where the sound energy goes away.

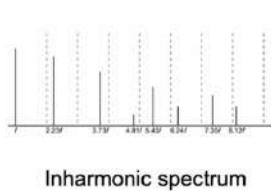
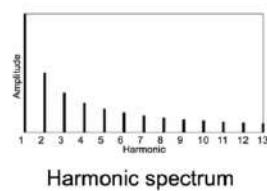


Since the transient has to do with chaotic phenomena (quite complicate to model) when we synthesise a sound we focus just on the main parts. It is interesting to notice that transients helps the recognition of the instrument: this means that our perception of sound is very detailed.

## Timbre

The timbre is a collection of all the aspects of the sound that allows us to identify the instrument that is playing. But, from an operative point of view, which are the features that make the timbre?

- Frequency content (spectrum)
- Transients (For example, if we remove the transient from the soundwave produced by an oboe, it becomes very hard to distinguish it from a human voice).
- Presence of and relation between partials, which can be in
  - o Harmonic ratio (for example the one produced by a string)
  - o Enharmonic ratio (non-multiple relations as it happens for percussions). For instance, the sound of a Gong and a Tam-Tam (Gong without something) is chaotic. Furthermore, we know that the harmonic relation is lost if we go from mono dimensional system to multidimensional one.



So according to that the sound can be classified as tonal or noisy.

## Acoustic waves

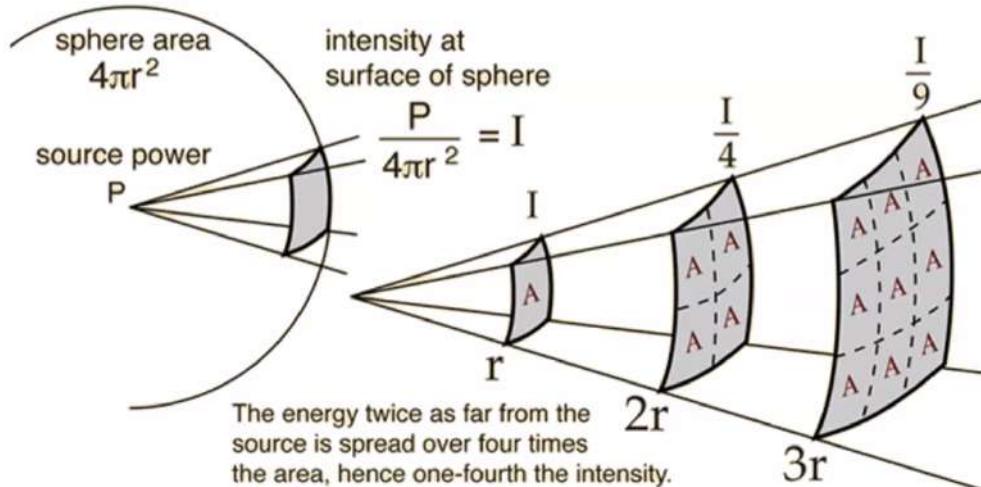
Acoustic waves are modification of air pressure over time. We can describe the waveform using the inverse of the period, called frequency. If we look at the waveform as a function of space, we can describe it in terms of wavelength  $\lambda$ .

Given the fact that the soundwave is propagated through a medium, the speed of sound increases with temperature as the temperature changes the properties of the material. It changes also with the type of material, that's why when we inhale helium we sound as a duck.

When a wave hits an object, there are a series of phenomena that can happen:

### 1. Diffusion and Attenuation

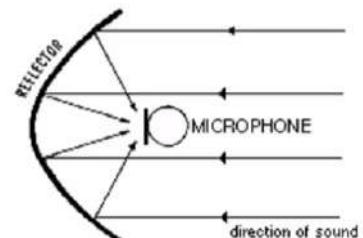
Due to propagation, spherical waves get attenuated in space. Simply speaking, as we go further from the source, a same amount of energy has to "move" a larger surface.



### 2. Reflection

It occurs anytime a sound wave impinges against an object bigger than its wavelength. This effect can generate peculiar effects such as:

- Reverb: this effect is generated when a sound bounces multiple times on a wall.



One interesting thing about the reverberation time in regular ambient is proportional to the volume of the environment. It is described by the Sarnies equation.

The time that sound needs to decrease of 60dB is equal to

$$T_{60} = 0,161 \cdot \frac{V}{\sum_i \alpha_i S_i} [s]$$

It is a workplace when the  $T_{60} \geq 4 \text{ } \% 5 \text{ sec}$ .

Bad effects: it increases the environment noise; therefore, it needs to be kept under control in the design of Lo-Fi environments (workplace, classrooms, industrial spaces, etc.).

Good effects: it constitutes an important clue to human sense of balance and orientation (greatly diminished in an anechoic chamber). In Hi-Fi rooms, we often try to boost reflection in order to achieve reverberation of at least 1s. A good design seeks a type of reverberation that adds *spatiality*, *warmth* and *coverage* to sounds. Digital techniques allow us to control reverberation in a post-production phase or in real time (live-electronics) to

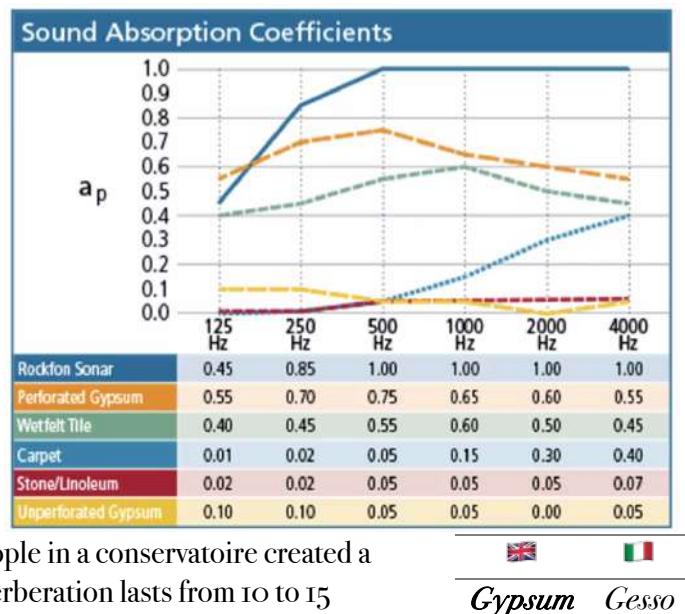
simulate different environments. Four examples can be: bright, dark, warm and closed environments. Blinded people can recognise the dimension of a room just by the reverberation of their voices. At Bovisa there is an anechoic room under the energetic CSS.

- Echo: a reflection becomes an echo when the time of reflection exceeds 50 ms that is the minimum time that our auditory system needs to perceive two sounds.

It can be exploited to focus a sound field. This what happens with parabolic reflector.

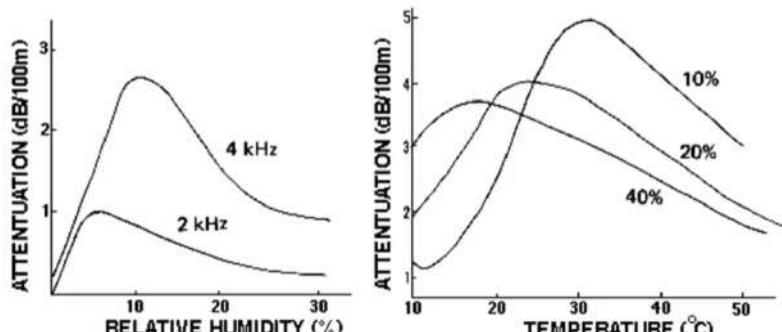
### 3. Absorption

Absorption can be defined as the loss of energy through some material. What is not absorbed is transmitted or reflected. To quantise the amount of absorption, we introduce the absorption coefficient  $\alpha$  whose range goes from 0 to 1. It is usually expressed as a function of frequency, and it is strongly dependent on the acoustic insulation of the material. It can be accurately measured thanks to several specialised techniques. It is important in the design of concert halls, theatres, recording studios, etc.... The extension of the absorption coefficient to an actual piece of a certain material is the total absorption. It is measured in Sabines, and it is calculated by multiplying the coefficient of absorption by the area of material in square feet. Fun fact: crazy people in a conservatoire created a room covered with gypsum where the reverberation lasts from 10 to 15 seconds.



### 4. Atmospheric effects

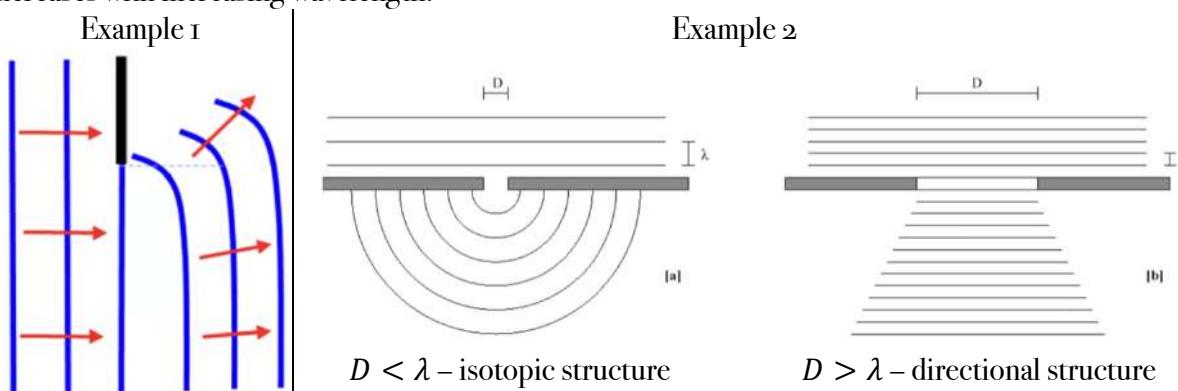
The absorption also depends on the air characteristics: temperature and humidity. The following graphs show how absorption vary in function of humidity and temperature.



Let us dwell one minute more on how temperature affects sound: the propagation speed increases with temperature. When temperature increases with altitude (for example in sunny winter days), sound is reflected downward, and propagation follow the earth curvature. Furthermore, sound is also affected by wind: due to turbulence, wave propagating in the direction of the wind tend to be reflected downward; those against the winds go upward.

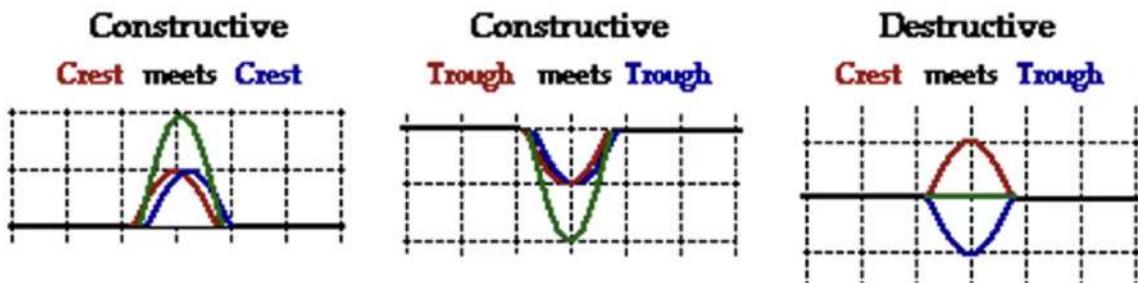
### 5. Diffraction

Diffraction is the bending of waves around obstacles and openings. The amount of diffraction increases with increasing wavelength.



### 6. Interference

Two or more waveforms that propagate at the same time produce a pressure that is the algebraic sum of pressure changes of each component. Considering sound waves coming from different sources: where wave maxima meet, we have constructive interference; while where wave maxima meet with wave minima, we have destructive interference.



### 7. Doppler effect

We can experiment the Doppler effect each time there is some kind of movement with sound. There are four possible scenarios:

- i. The source is moving to the observer, and the observer is firm.

In this situation, every front wave is chased by the next one since the next one “starts” some meters ahead. It is like if the sources chased the front waves. We model this situation considering that a generic wavelength can be seen as the space that the wave has travelled in a period:

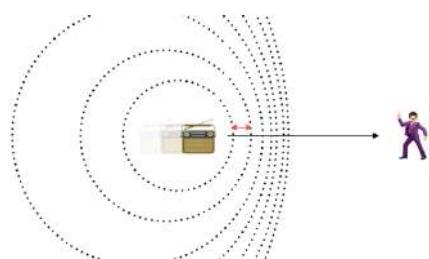
$$\lambda = cT$$

If the source is moving, the wavelength has to be adjusted because the next front wave will be shifted by a portion of space equals to the distance that the source has swept. When the source is moving, a generic front wave is closer to the next one compared to the situation in which the source is still. So, the new wavelength will be shorter than the original (as we can see from the picture).

$$\lambda' = cT - v_0 T$$

From this expression, we find the frequency.

$$\frac{c}{f'} = T(c - v_0) \rightarrow f' = \frac{c}{c - v_0} f$$



The frequency perceived increases due to the doppler Effect since the original frequency is multiplied by a factor bigger than one.

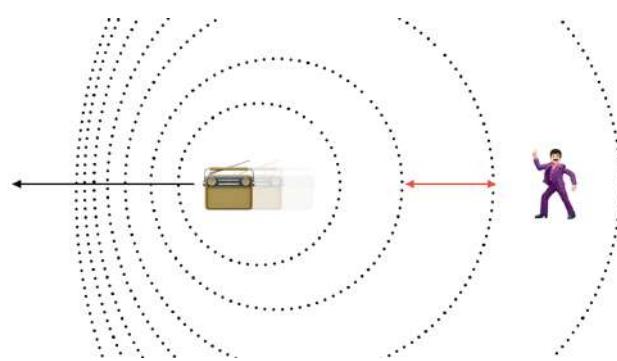
2. The source is moving away from observer, and the observer is firm.

This is the opposite situation: the wavelength is longer than the one that we had when the source is still.

Each wavefront is more distant to the previous one since the source is moving away. The new wavelength will be

$$\lambda' = cT + v_0 T$$

From which we obtain



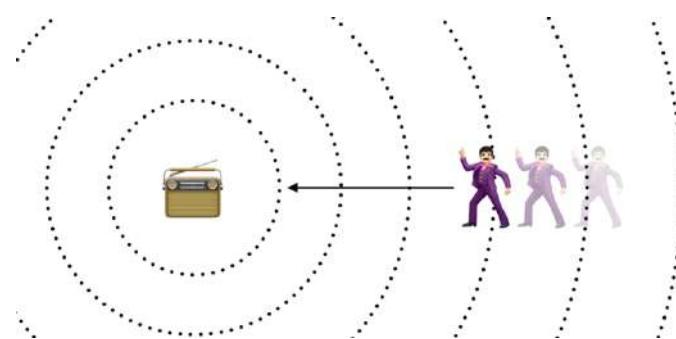
so a frequency that is perceived lower than the original since the ratio is smaller than one.

3. The source is still, and the observer is moving to it.

This situation is easier to imagine. This is exactly the same situation we experimented when we run towards the sea waves, when we were children. The next wave impacts against ourselves "earlier" if we move towards it. So, the distance between two waves seems shorter:

$$\lambda' = cT - v_0 T$$

From which we obtain:  $f' = \frac{c}{c-v_0} f$



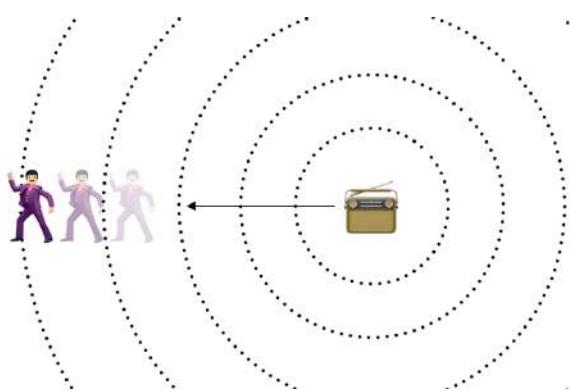
4. The source is still, and the observer is moving away from it.

Exactly as before, but we are running away from the waves. As if we are chasing the wave from the open sea to the shoreline. It is like when we are surfing. If we are fast enough, it seems that the second wave does not come at all. So the distance between two wavefronts seems bigger

$$\lambda' = cT + v_0 T$$

From which we obtain

$$f' = \frac{c}{c+v_0} f$$



## Psychoacoustics

### The Ear

#### Outer Ear

The outer ear is made up by the pinna and the ear canal. The first one is very peculiar for each person such that it could be considered as unique as a fingerprint. According to that, every man hears differently from any other man.

The ear canal is very deep, it arrives up to under the eyes.

#### Middle Ear

The ear temporal bone that protects the inner and medium ear is the hardest part of the whole body. The eustachian tube connects the ear canal to the throat. It equalises the pressure between the inner and the external part of the year protecting the tympanum from violent noises and changes in air pressure. The bones that connect the ear drum to the oval window are called hammer, anvil and stirrup. They work like an electrical transformer: they amplify the pressure and reduce the motion preserving the same energy. They adapt the sound wave, that is travelling in the air, to the liquid contained in the cochlea.

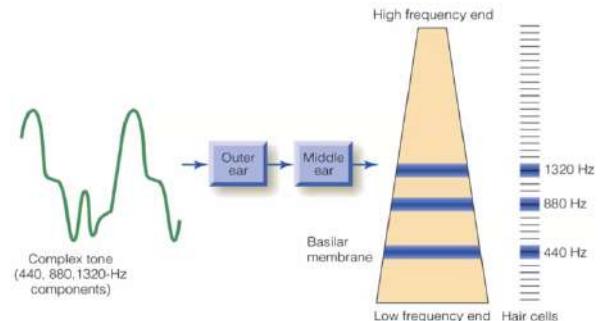


electrical transformer: they amplify the pressure and reduce the motion preserving the same energy. They adapt the sound wave, that is travelling in the air, to the liquid contained in the cochlea.

#### Inner Ear

The most important part of the inner ear is undoubtedly the cochlea. It is a labyrinthic structure where the vibrations are converted into electrical signals. It is filled with a liquid called perilymph that has an ionic composition close to the one of plasma and cerebrospinal fluid. The conduct of the cochlea is divided in two by the basilar membrane throughout which are more than 30,000 hair cells (the ones that, once they are damaged, provoke tinnitus) that convert the wave into neural pulses. Each region of the basilar membrane is maximally sensitive to a different frequency. The resonance frequency doubles every 3.5 mm that means that the space dedicated to every frequency follows a logarithmic function:

$$\omega_0 = 2^{\frac{x}{3.5}} + \alpha \Rightarrow x = \frac{\log_2(\omega_0 - \alpha)}{3.5}$$



Where  $x$  is the distance between the oval window and the site of the membrane that is sensible to the frequency  $\omega_0$ . So, the space between two frequencies is

$$\Delta x = x_2 - x_1 = \frac{\log_2(\omega_{0_1} - \alpha)}{3.5} - \frac{\log_2(\omega_{0_2} - \alpha)}{3.5} = \frac{\log_2\left(\frac{\omega_{0_1} - \alpha}{\omega_{0_2} - \alpha}\right)}{3.5}$$

The lower is the frequency, the deeper will be stimulated the cochlea. The cochlea has two windows: the oval (that is connected to the stirrup) and the round. The round window works as a dumping mechanism that compensates for the alteration of pressure generated by the oscillation of the oval window. Cells that are close to the oval window will perceive the vibration earlier than the ones that are at the end. But we cannot perceive any delay! This is why our brain compensates it.

### Critical band

The critical band is the range of frequencies that activate the same hair cell. It basically described the ear's ability to separate simultaneous sounds. The whole audible spectrum is covered by 24 critical bands of about 1/3 of octave. We have to underline that if we have some hearing losses, we have not too many cells to replace them. So, pay attention to protect our ears.

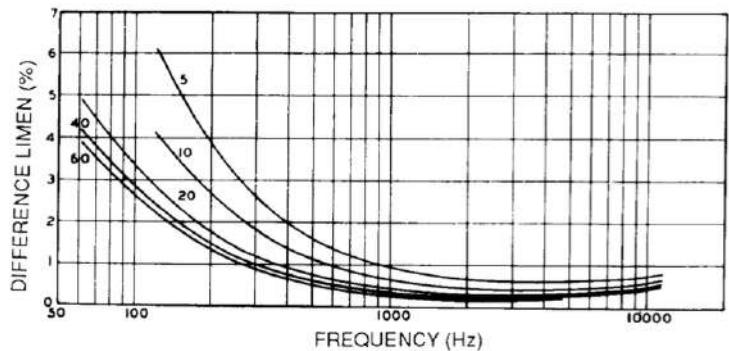
### Pitch

We define pitch as the subjective impression of frequency allowing us to identify a certain sound linking it to note on a musical scale. Being a psychoacoustic variable, it depends on subject sensitivity (for some lucky people who have perfect pitch, it is stored in mind as it is the language). It is possible to discriminate between 1400 different levels of pitch, among which 120 are commonly used in occidental scales.

We define the Just noticeable difference (JND) as the smallest variation of frequency we can perceive. As we can imagine by considering what we have just said about the perception of sound, it depends on frequency range and sound intensity. It is quite interesting to notice how the frequency range in which we can hear better is the speech area (1-4 KHz).

The limits in our acoustic perception are:

- Lower bound: 20-30 Hz
- Upper bound: 15-20 KHz (but it depends on how long an individual was exposed to noises and loud sounds)



The pitch perception is connected to the length of the signal. If the length is shorter to 15-20 ms we are not able to recognise the pitch and we perceive it as an impulsive noise

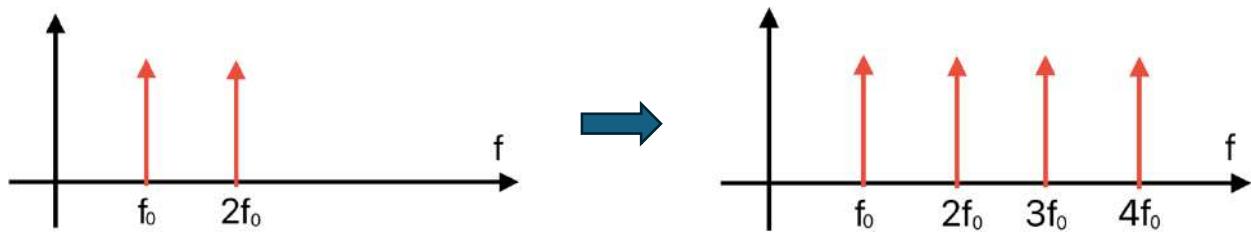
### Mathematical structure of a sound

Almost all sounds that give a sensation of pitch are periodic. Their spectrum consists of harmonics that are integer multiples of the fundamental (Fourier series). The pitch of a complex periodic tone is close to the pitch of a sine wave at the fundamental. Helmholtz claimed that the pitch is heard at the fundamental since the fundamental frequency gives the lowest frequency peak on the basilar membrane. Related to that, we can say something about the *missing fundamental*: Seebeck (and later Schouten) showed that complex periodic sounds with NO energy at the fundamental may still give a clear pitch sensation at the fundamental (for example, telephone speech: the telephone acts as a high-pass filter, removing energy below about 300 Hz). Why? The harmonic components interact with each other and, as a result, a fundamental frequency appears. Let us see it with an example

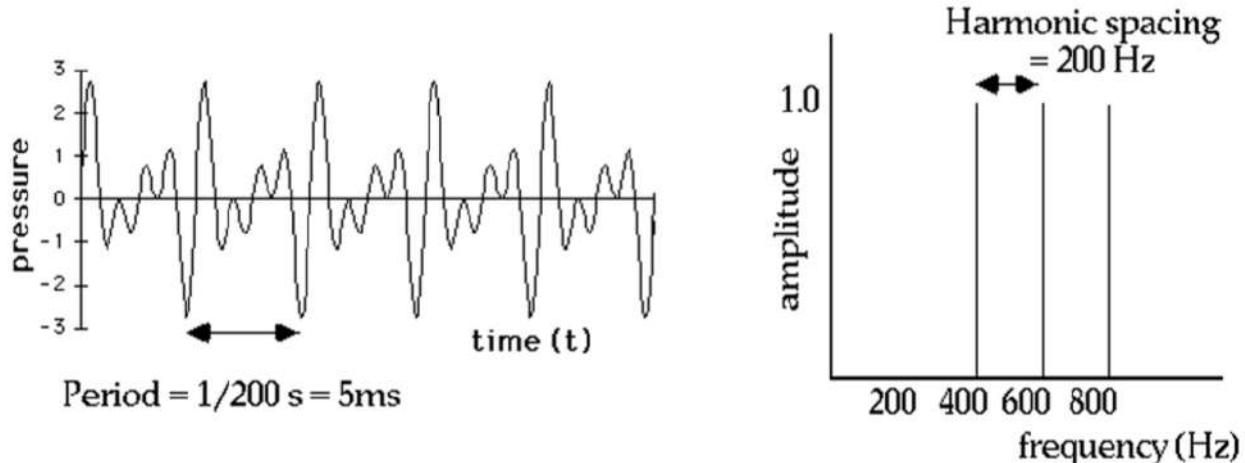
$$x(t) = \underbrace{\sin(2\pi f_0 t)}_{\text{fundamental}} + \underbrace{\sin(2\pi 2f_0 t)}_{\text{first harmonic}}$$

Let us apply a non-linear filter, for example the one that compute the square value

$$\begin{aligned} y(t) &= x^2(t) = (\sin(2\pi f_0 t) + \sin(2\pi 2f_0 t))^2 = \\ &= \sin^2(2\pi f_0 t) + \sin^2(2\pi 2f_0 t) + 2 \sin(2\pi f_0 t) \sin(2\pi 2f_0 t) \end{aligned}$$



When the fundamental is missing, it is evaluated by the cochlea (actually, it is simply the result of a mathematical operation between the waves in the air, the cochlea does not do any maths) as the difference between the second and the first harmonic.



Also related to that, we have to underline that we need an observation time of at least two periods to perceive that a certain sound is actually a sound and not noise. In fact, if we hear the sound of a half wave, we perceive it as a click independently from its frequency.

### *Pitch and intensity and pitch and frequency*

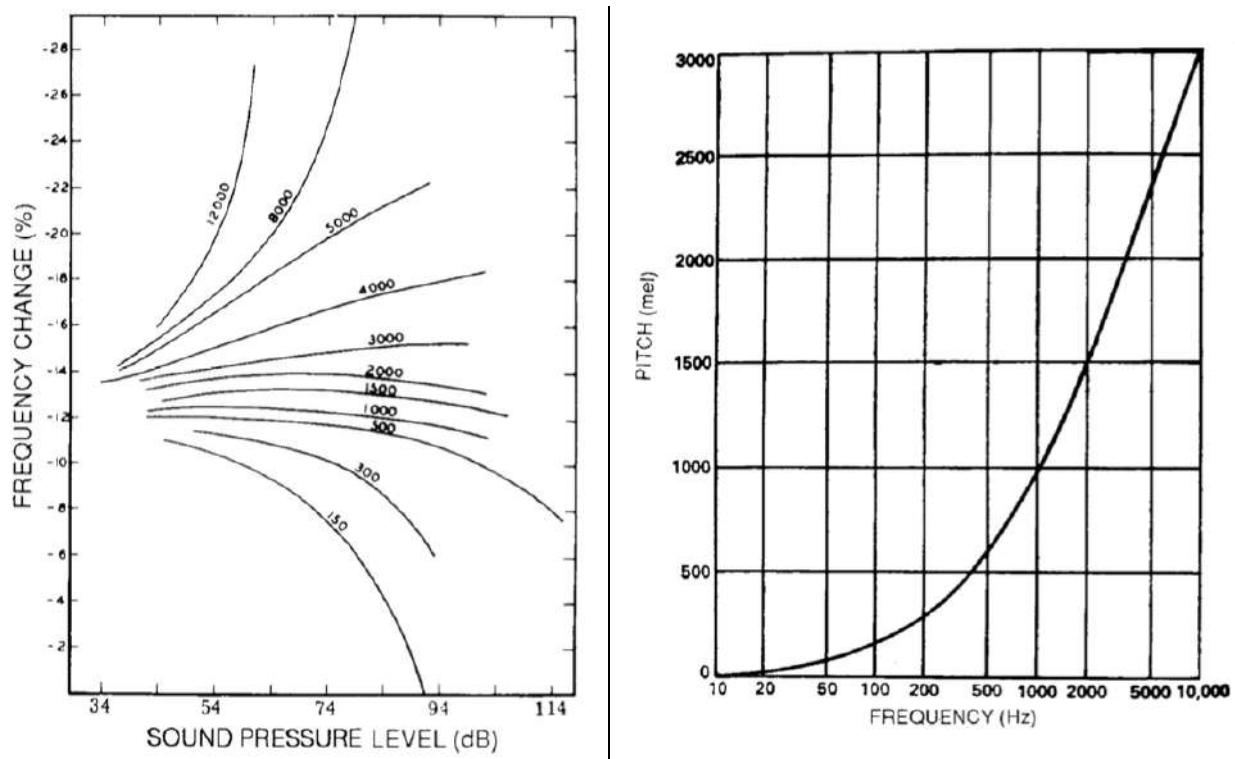
As we said before, pitch is a function of the intensity. This means that a same sound (same frequency) is perceived as a different note (different pitch) if it is played with different intensities.

We can observe how the perceived frequency changes in percentage when we change SPL in the chart below. Professional singers have to be really aware of that in order to be precise both in piano and in forte.

How can we read it?

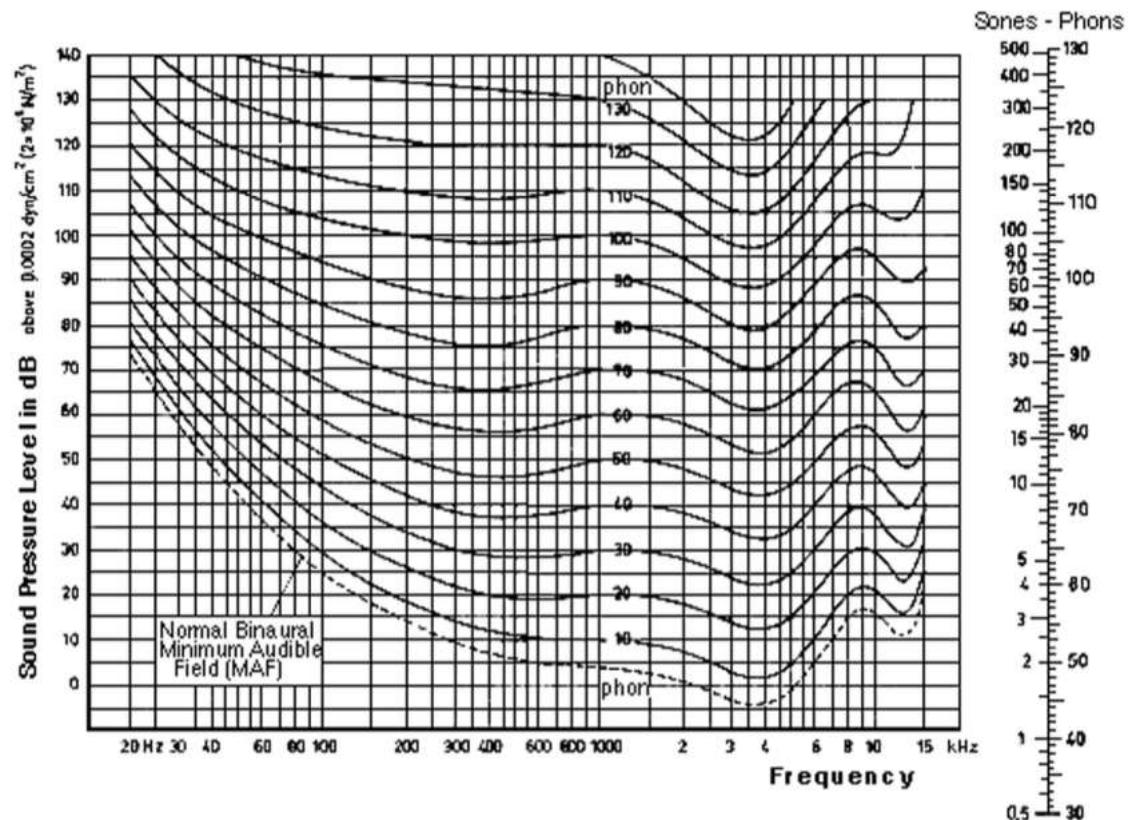
A descending curve means that frequency must increase in order to keep the perceived pitch constant.

The most important relation that links the pitch with an objective and quantitative variable is the one with frequency. As we can see below, the trend recalls the one of an exponential function. This means that the lower are two notes, the higher will be the difference in frequency between them if we compare between the same two notes at higher octaves.



### *Acoustic intensity measurements*

We define **loudness** the subjective impression of sound intensity and we measure the loudness level in **Phon**. We define, in turn, the **Phon** as the value of the isophonic curve (by Fletcher and Munson) correspondent to a certain SPL and to a certain frequency.



We can see how also the hearing and pain threshold are function of the frequency. Furthermore, we can observe that the upper bound of the audible frequency range decreases as the age increases.

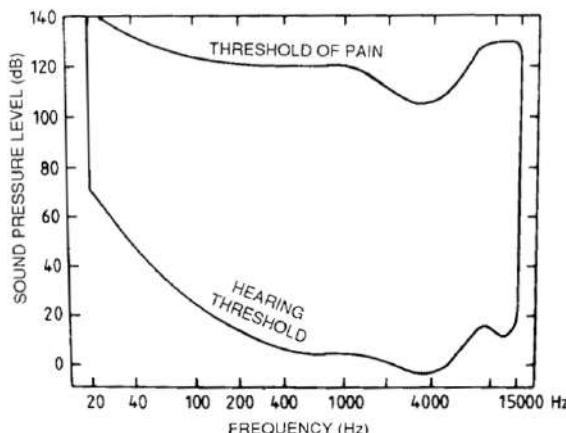


Figure 3.8 Audible frequency and sound level range.

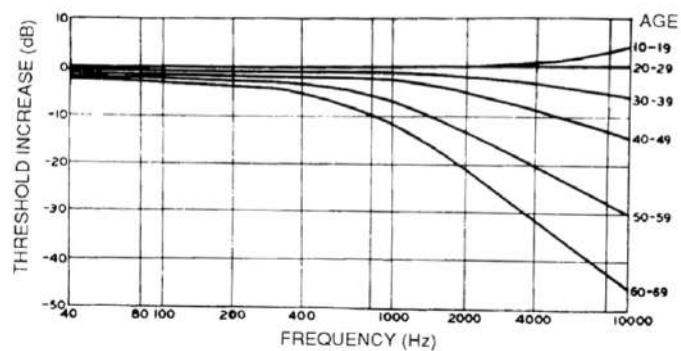


Figure 3.10 Dependence of the hearing threshold on age. (From Olson, 1968.)

## Threshold shift

It could happen to experiment the so-called **threshold shift**. This is nothing but the formal definition of the sensation of deafness we experiment when we are subjected to high volume noises. The level of the hearing loss depends on age and gender: males are most sensitive at low frequencies while females are more sensitive at high frequencies. In general, the most damaging noises are between 2 and 4 KHz. But why do we temporary lose hearing? Because the noise causes blood vessels to shrink, which reduces blood intake to hair cells. It is similar to what happens to eyes when going from a brightly room to a dimly lit room.

There exists two main type of threshold shift classified according to the duration of the phenomena:

1. Temporary Threshold Shift (TTS):

We need from 24 hours to a week to recover the previous level of hearing.

2. Permanent Threshold Shift (PTS):

After being exposed to the noise for long time, sometimes we cannot anymore recover the previous threshold. We use the expression “Chronic Threshold Shift” when TTS turns into PTS. This could happen when we are frequently exposed to noise without having time to recuperate.

Furthermore, the hair cells die as time goes by and they are not renewed. The two categories more exposed to hearing losses are pneumatic drill operator and violin players.

## Masking

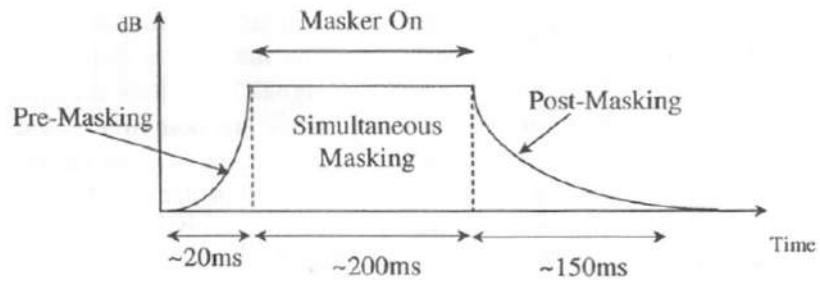
### Frequency masking

A certain tone that stimulates the same hair cells of other tones lower in intensity cover them. This means that we are not more able to perceive them. This process is called frequency masking.

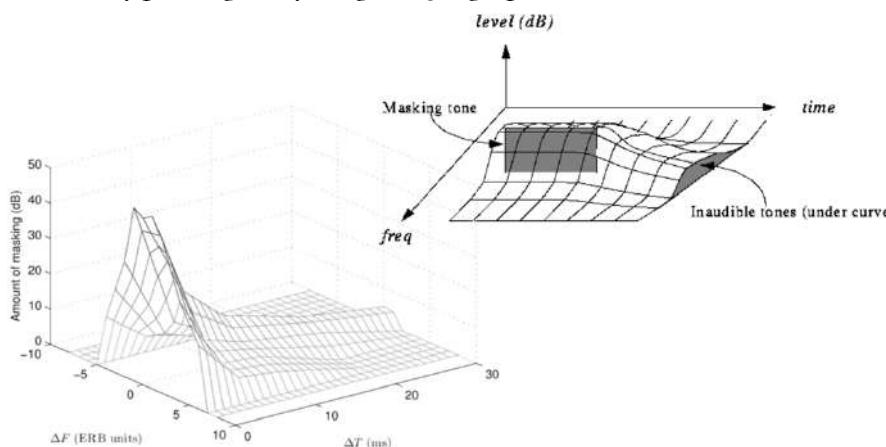
### Temporal masking

The same happens in time domain:

after being exposed to a louder sound, the ear needs some time to be able to perceive a softer sound (of similar frequency). The louder tone will mask softer adjacent tones within a certain variable time threshold. What it is surprising is to discover that there is not only a post-masking but also a pre-masking. We suppose, because there is a sort of lag-time between the response of the tympanum and the response of the brain (the neural connections). Like if we forget the previous one.



To visualise a complete representation of the phenomena, we can put together both the information about frequency and about time by plotting everything in a 3D graph. This is what we obtain.



## Acoustics effects

### First-order effects

The first order effects are that kind of acoustic effects generated inside the cochlea. One of the most famous is the **beats**. We can hear beats each time we have two different sounds whose frequency are slightly different one with the other. What we perceive is a single sound at frequency  $f = \frac{f_1 + f_2}{2}$ , whose amplitude is modulated at half the frequency deviation. The simplest way to visualise this process is just applying maths.

$$\sin(2\pi f_1 t) + \sin(2\pi f_2 t) = \underbrace{2 \sin\left(2\pi \frac{f_1 + f_2}{2} t\right)}_{\text{Tone Frequency}} \underbrace{\cos\left(2\pi \frac{f_1 - f_2}{2} t\right)}_{\text{Modulation (beat) frequency}}$$

Another effect generated in the cochlea is the **combination sounds**. By superposing a series of pure tones, we perceive notes that are not present in the frequency we are reproducing.

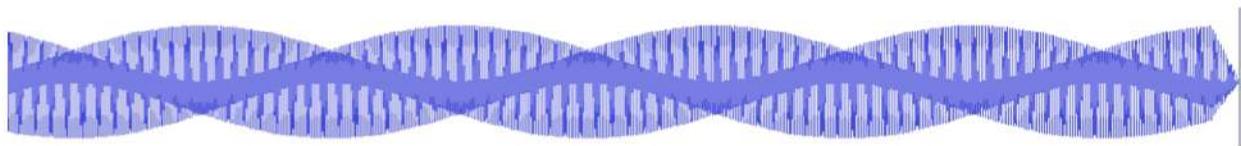
Furthermore, we can cite the **Risset's musical beats**. If we play a series of simultaneous sounds characterised by

- 7 Harmonics of magnitude  $\frac{1}{f}$
  - fundamental frequencies that are slightly different one from the other
- we hear a sort of “phaser effect” with a rhythm that resembles the 7/8.

### *Second-order effects*

The second-order effects are that kind of effects originate in the neural system. One example is given by the perceived beats generated by two sounds at frequency  $f_1$  and  $f_2 \cong 2f_1$ . We hear a tone beat equals to  $f_2 - 2f_1$  due to the cyclic changes in the waveshape (if  $f_2$  is not exactly equals to  $2f_1$  the delay makes the two waveshape not to match perfectly and become as dephased as time goes by).

The result is the one we can see in the following image:

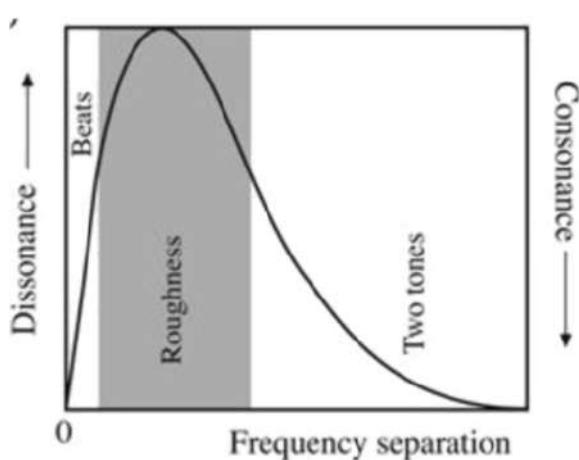


Another effect is the **reconstruction of the fundamental**: when we play a series of pure sound like  $2f, 3f, 4f, \dots$  the pitch we perceive is the one related to the absent fundamental frequency  $f$ .

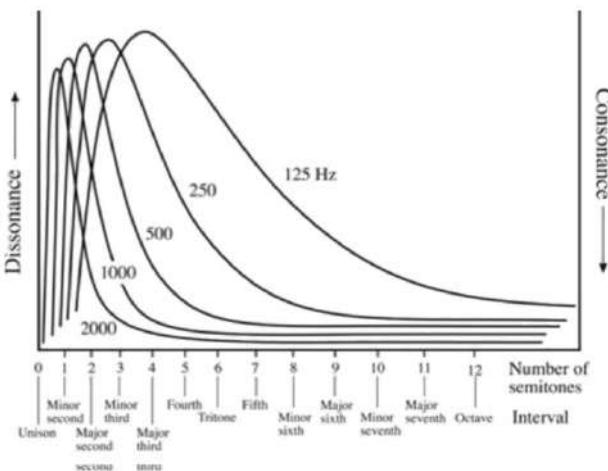
### **Timbral consonance**

The concept of timbral consonance is closely linked to the concept of critical bandwidth. Given a certain tone at a certain frequency  $f$ , the tones that fall inside the critical bandwidth are the ones that activate the same hair cells in the cochlea. The relation between frequency and dissonance is explained in the following graphs:

When the rate of musical beats exceeds about 20 Hz, they create an audible sensation of roughness. Once the frequency separation is large enough, we perceive two separate tones.



The width and the position of the peak in the band of sensory dissonance depends on the absolute frequencies of the tones. This means that an interval that might be consonant in a high register, high up on the piano keyboard, may become dissonant in a lower register. This is why piano players “does not” sound chords with the left hand.



In other words, *there is no such thing as a tonally dissonant interval* – it all depends on where you play it, using high notes or low.

Some examples:

- *In the mid-range of the piano, intervals of a minor third (three semitones) generally lie beyond the band of roughness, evading sensory dissonance. For high notes, even a semitone (minor second) interval does not create roughness. But in the bass, rather wide intervals can become rough, and even an interval like a perfect fifth that is considered highly musically consonant becomes dissonant in sensory terms. This explains the ‘gruffness’ of chords sounded low down with the left hand and helps us understand why Western music almost universally shows voicings (combinations of notes) that become more widely spaced the lower they are sounded.*
- *The left hand of a pianist will typically play voicings spanning almost an octave or more when the lowest note is around the octave below middle C, while it might include some fourths and fifths if this note is higher than the E below middle C. The right hand, meanwhile, merrily bangs out chords containing thirds and even seconds.*

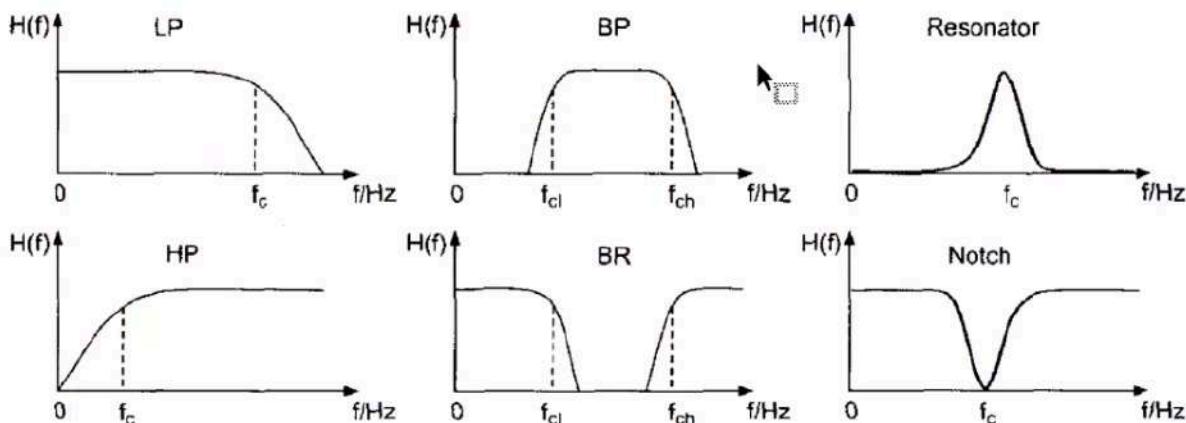
## Digital Audio Effect

Digital Audio Effect are nothing but software able to change sound according to certain parameters that the user can shuffle.

There are plenty of digital audio effects. They can be divided in three main classes:

- Audio equalisers
- Integer and fractional delay filters
- Delay-based effects

Audio equalisation is the process of altering the frequency content of a sound by means of linear filters. They are mostly used to better shape the frequency response of a sound in studio recording or to adapt the sound to a certain environment. We report a brief taxonomy in a table the filter classes.



Low pass and high pass filter reject all the frequencies that are over or below a certain cut off frequency.

Bandpass and band-reject filters select a portion of the frequency content that has to pass or that it has to be removed.

The resonator and the notch filters are like band pass and band reject filters but shaped for a specific frequency.

Filter classification:

- **Lowpass:** attenuate frequencies higher than  $f_c$
- **Highpass:** attenuate frequencies lower than  $f_c$
- **Bandpass:** attenuate frequencies outside the frequency band  $[f_{cl}, f_{ch}]$
- **Bandreject:** attenuate frequencies inside the frequency band  $[f_{cl}, f_{ch}]$
- **Notch:** attenuate frequencies in a narrow frequency band around  $f_c$
- **Resonator:** amplify frequencies in a narrow frequency band around  $f_c$
- **Allpass:** pass all frequencies but modifies the phase

We notice that there is also a particular class called **allpass**. This class of filters lets untouched the magnitude of the spectrum of a signal but modifies the phase.



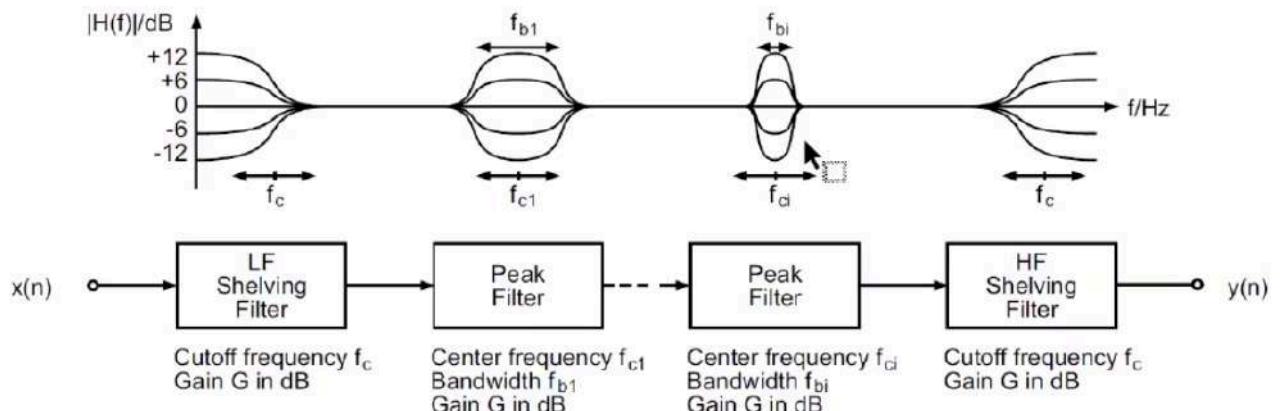
[To shelve](#)

[Digradare](#)

All the filters we saw can be included in higher order classes that are:

- **Shelving filters:** based on lowpass/highpass filters and a direct path. Parametrized by cutoff frequency and gain factor.
- **Peaking filters:** based on bandpass filters and a direct path. Parametrized by centre frequency, bandwidth and gain factor.
- No stop band.

Each kind of filter can be implemented as a differently parametrised cascade of these three classes of filters.

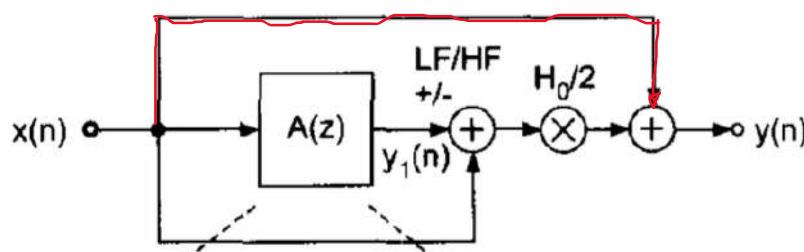


Where LF range is equalised by a lowpass shelving filter, the HF range is equalised by an highpass shelving filter and the MF range id equalised by a series of two peak filters.

Let us now go through each classes of filters.

### First-order shelving filters

Block scheme



Transfer function:

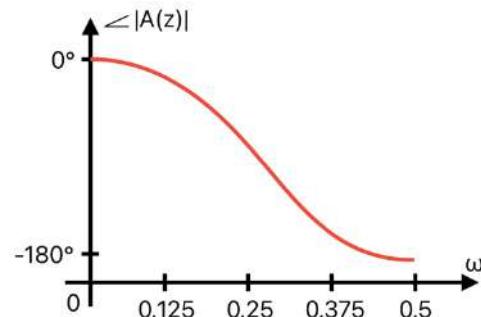
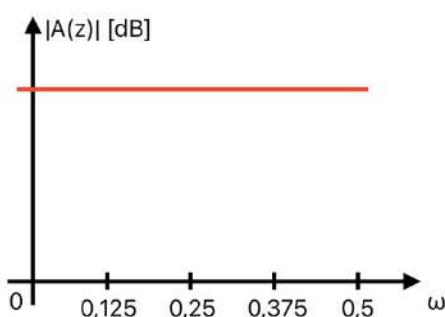
$$H(z) = 1 + \frac{H_0}{2} [1 \pm A(z)]$$

Where:

$$A(z) = \frac{a + z^{-1}}{1 + az^{-1}}$$

is a first-order allpass filter.

The Plus/minus sign correspond to low/high frequency shelving. Let us see why changing the sign, we change the type of filter. If we plot the magnitude of this filter we get a flat response. While the phase goes from 0 and to  $-180^\circ$ .



If we sum the filter contribution to the direct path, we get a phase cancellation of high frequencies since they will be summed to the opposite ( $-180^\circ$ ) of themselves. On the other hand, if we use the minus sign we remove the low frequencies due to the same reason.

But why using this kind of filters? Because they are parametric, by simply setting some parameters we can obtain the response we want:

- Gain  $G_{dB}$  is adjusted by

$$H_0 = V_0 - 1, \quad V_0 = 10^{\frac{G_{dB}}{20}}$$

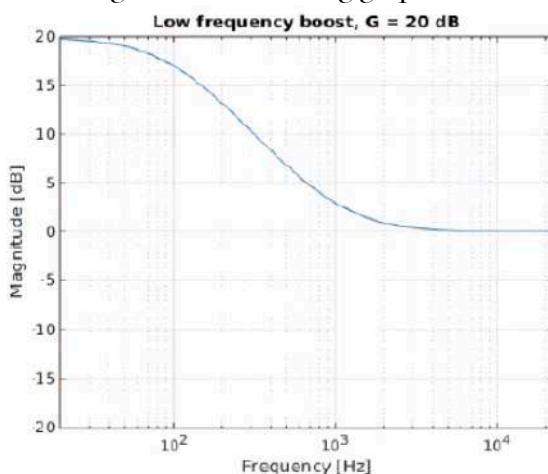
- Low frequency shelving ( $\Omega_c = 2\pi f_c/f_s$ )

$$a = \begin{cases} \frac{\tan\left(\frac{\Omega_c}{2}\right) - 1}{\tan\left(\frac{\Omega_c}{2}\right) + 1}, & \text{for } G_{dB} \geq 0 \\ \frac{\tan\left(\frac{\Omega_c}{2}\right) - V_0}{\tan\left(\frac{\Omega_c}{2}\right) + V_0}, & \text{for } G_{dB} < 0 \end{cases}$$

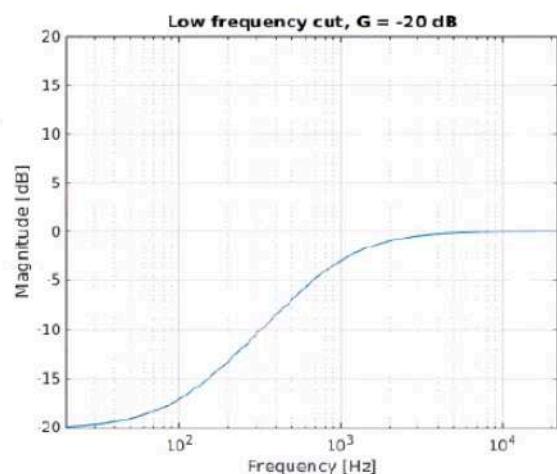
- High frequency shelving ( $\Omega_c = 2\pi f_c/f_s$ )

$$a = \begin{cases} \frac{\tan\left(\frac{\Omega_c}{2}\right) - 1}{\tan\left(\frac{\Omega_c}{2}\right) + 1}, & \text{for } G_{dB} \geq 0 \\ \frac{V_0 \tan\left(\frac{\Omega_c}{2}\right) - 1}{V_0 \tan\left(\frac{\Omega_c}{2}\right) + 1}, & \text{for } G_{dB} < 0 \end{cases}$$

If we try to plot on Matlab the magnitude of a low frequency shelving with  $f_c = 100$  Hz and different gains, that is what we get are the following graphs.



(a) Boost,  $G_{dB} = 20$



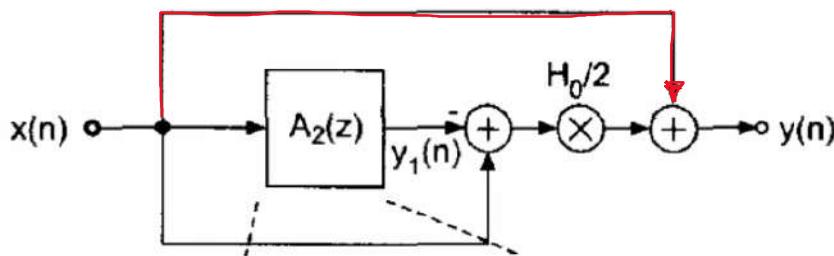
(b) Cut,  $G_{dB} = -20$

## Peak filters (or second order filters)

As we can see, the structure is pretty similar

### Block scheme

### Transfer function:



$$H(z) = 1 + \frac{H_0}{2} [1 - A_2(z)]$$

Where:

$$A_2(z) = \frac{-a + d(1-a)z^{-1} + z^{-2}}{1 + d(1-a)z^{-1} - az^{-2}}$$

is a second-order allpass filter.

In this case the phase goes from 0 to  $-360^\circ$ . Around to  $-180^\circ$  we have the cancellation part of the filters, as the frequency grows, we have a coherent summation. That is why we get a band pass filter.

This time again, we have some parameters that we can set:

- Gain  $G_{dB}$  is adjusted by

$$H_0 = V_0 - 1, \quad V_0 = 10^{\frac{G_{dB}}{20}}$$

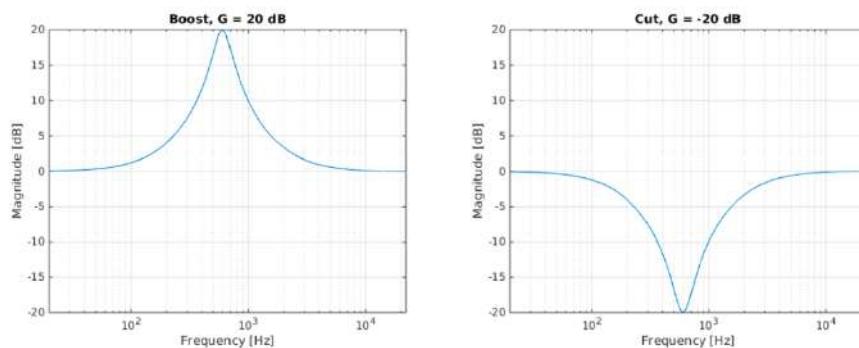
- Centre frequency parameter ( $\Omega_c = 2\pi f_c/f_s$ )

$$d = -\cos(\Omega_c)$$

- Bandwidth parameter ( $\Omega_b = 2\pi f_c/f_s$ )

$$a = \begin{cases} \frac{\tan\left(\frac{\Omega_b}{2}\right) - 1}{\tan\left(\frac{\Omega_b}{2}\right) + 1}, & \text{for } G_{dB} \geq 0 \\ \frac{\tan\left(\frac{\Omega_b}{2}\right) - V_0}{\tan\left(\frac{\Omega_b}{2}\right) + V_0}, & \text{for } G_{dB} < 0 \end{cases}$$

If we try to plot on Matlab the magnitude of a peak filter with  $f_c = 600 \text{ Hz}$ ,  $f_b = 100 \text{ Hz}$  and different gains, that is what we obtain two graphs like the following ones.



(a) Boost,  $G_{dB} = 20$

(b) Cut,  $G_{dB} = -20$

The part related to audio equalisation is concluded, we now move to

### Integer and fractional delay filters – FIR Filters

We are now interested to implement delay in a signal. Mathematically the delay is expressed by a shifting in time domain:

$$y(t) = x(t - \tau) \quad \tau \in \mathbb{R}$$

In discrete time domain, we can only shift of integer numbers if we do not want to break the math behind it. By considering  $n = tf_s$ , we get

$$y(n) = x(n - D)$$

Quite often, we need a high-resolution delay maybe to delay the signal of a time in between two samples. (con che inglese l'ho scritto) In order to do it, we introduce a fractional delay that allows us to delay a discrete-time signal by a non-integer delay

$$D = \lfloor D \rfloor + d, \quad d \in \mathbb{R} | 0 < d < 1$$

However, the formula  $y(n) = x(n - D)$  holds only for integer values of  $D$ . If  $D \notin \mathbb{N}$ , output value lies in between two samples. Appropriate value must be found via **band limited interpolation**. We can look at the delay problem as a **resampling** problem divided in two steps:

- Reconstruct the bandlimited continuous-time signal
- Resample it after shifting

Formally, if we do not break the sampling theorem, we can convert a digital signal into continuous one by reconstruct them from its samples with a interpolation

$$\hat{x}(t) \triangleq \sum_{n=-\infty}^{\infty} x(nT_s) h_s(t' - nT_s) \equiv x(t)$$

$$\text{Where } h_s(t) = \text{sinc}(f_s t) \triangleq \frac{\sin(\pi f_s t)}{\pi f_s t}$$

What is true is that reconstruction and resampling can be approximated by linear filtering. Let us see how we can set the problem as a linear filtering problem.

Given the shift in time  $y(n) = x(n - D)$ , we know that its Z-transform is equal to

$$Y(z) = \mathcal{Z}\{y(n)\} = z^{-D} X(z)$$

Therefore, the transfer function we need to implement a delay will be

$$H(z)|_{z=e^{j\omega}} = \frac{Y(z)}{X(z)} = z^{-D}$$

that in the omega domain it becomes

$$H_{id}(\omega) = e^{-j\omega D}$$

Where:

- $\omega = \frac{2\pi f}{f_s}$  is the normalised angular frequency
- Magnitude:  $|H_{id}(\omega)| = 1$
- Phase:  $\angle H_{id}(\omega) = -\mathcal{D}\omega$

Delay can be characterised by two parameters:

- Group delay

$$\tau_g(\omega) = -\frac{\partial \angle H_{id}(\omega)}{\partial \omega}$$

That describes the time shift of the envelope of a wave packet, a "pack" or "group" of oscillations centred around one frequency that travel together. Simply speaking, how many samples a specific frequency is delayed by the filter.

- Phase delay

$$\tau_p(\omega) = -\frac{\angle H_{id}(\omega)}{\omega}$$

For the frequency response  $H_{id}(\omega) = e^{-j\omega\mathcal{D}}$ ,  $\tau_{g,id} = \tau_{p,id} = \mathcal{D}$

The ideal solution with which we came up is the so-called discrete-time linear-phase allpass filter with unit magnitude and group delay equal to  $\mathcal{D}$ . Its impulse response is found via inverse DFT of

$$h_{id}(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} e^{-j\omega\mathcal{D}} e^{j\omega n} d\omega = \text{sinc}(n - \mathcal{D})$$

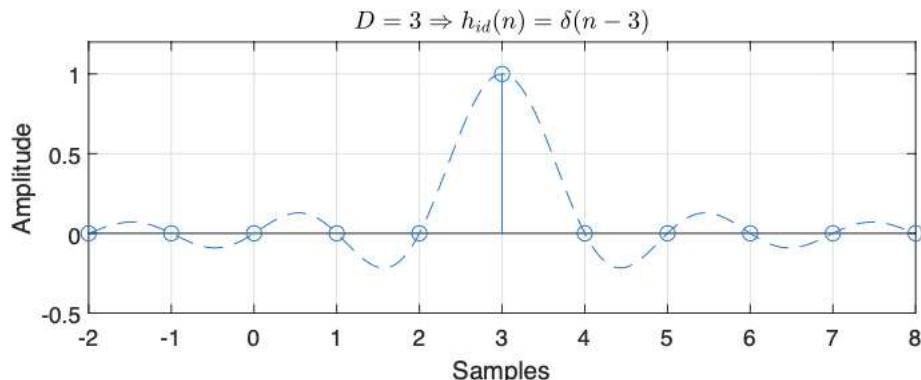
- If  $\mathcal{D}$  is integer,  $h_{id}(n)$  is a single impulse at  $n = \mathcal{D}$

$$h_{id}(n) = \delta(n - \mathcal{D})$$

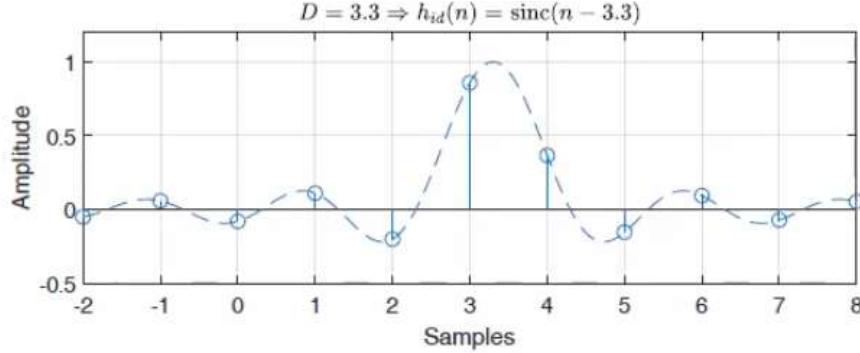
- If  $\mathcal{D}$  is not integer,  $h_{id}(n)$  is infinitely long and non-causal (hence, impossible to use in real-time context)

$$h_{id}(n) = \text{sinc}(n - \mathcal{D})$$

A sinc function looks like.



If  $D$  is not integer, by sampling a sinc we do not get just one impulse in  $n = 0$  but we have a series of impulses from  $-\infty$  to  $\infty$ .



This generates two problems: the function is non causal, and, furthermore, it is infinitely long. How can we cast this band limited interpolation problem into a linear problem?

A straightforward approach can be using FIR filters that could approximate the ideal fractional delay. The z-domain transfer function of  $N$ -th order FIR filter is

$$H(z) = \sum_{n=0}^N h(n)z^{-n}$$

Our goal is to find the coefficients  $h(n)$  that approximate  $h_{id}(n)$ . In order to proceed in this way, we define a cost function (an error) as the difference between the Fourier Transform of the coefficients and the Fourier Transform of the sinc we want to implement.

$$E(\omega) = H(\omega) - H_{id}(\omega)$$

There are many approaches based on FIR filters:

- A first one based on a least squared design;
- A second one, a windowing method;
- A third one called Lagrange Interpolation;
- A last one called GLS method.

### **Least-squares FIR**

The first approach we study is the least-square FIR. It is kind of a trivial approach: we define the error function in terms of Ordinary Least Squares (the L<sub>2</sub> norm): we minimise the  $L_2$  norm of the error function  $E(\omega) = H(\omega) - H_{id}(\omega)$ , defined as

$$\begin{aligned} \|E(\omega)\|_2^2 &= \frac{1}{2\pi} \int_{-\pi}^{\pi} |E(\omega)|^2 d\omega \\ &\stackrel{E(\omega)=H(\omega)-H_{id}(\omega)}{=} \frac{1}{2\pi} \int_{-\pi}^{\pi} |H(\omega) - H_{id}(\omega)|^2 d\omega \\ &\stackrel{\text{Parseval's Theorem}}{=} \sum_{n=-\infty}^{\infty} |h(n) - h_{id}(n)|^2 = \sum_{n=-\infty}^{\infty} [h^2(n) + h_{id}^2(n) - 2h(n)h_{id}(n)] \\ &= \sum_{n=-\infty}^{\infty} h_{id}^2(n) + \sum_{n=-\infty}^{\infty} [h^2(n) - 2h(n)h_{id}(n)] \\ &\stackrel{\text{Parseval's Theorem}}{=} \sum_{n} h_{id}^2(n) = \frac{1}{2\pi} \int_{[2\pi]} |H_{id}(\omega)|^2 d\omega \end{aligned}$$

$$\begin{aligned}
&= \frac{1}{2\pi} \int_{-\pi}^{\pi} |H_{id}(\omega)|^2 d\omega + \sum_{n=-\infty}^{\infty} [h^2(n) - 2h(n)h_{id}(n)] \\
&\stackrel{H_{id}(\omega)=e^{-j\omega\mathcal{D}}}{=} \sum_{n=-\infty}^{\infty} [h^2(n) - 2h(n)h_{id}(n)] \\
&= \frac{1}{2\pi} \int_{-\pi}^{\pi} |e^{-j\omega\mathcal{D}}|^2 d\omega + \sum_{n=-\infty}^{\infty} [h^2(n) - 2h(n)h_{id}(n)] \\
&\stackrel{\text{Symmetrical integral on the complex plane}}{=} \\
&= 1 + \sum_{n=-\infty}^{\infty} [h^2(n) - 2h(n)h_{id}(n)] \approx
\end{aligned}$$

If we are not convinced of the last passage

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} |e^{-j\omega\mathcal{D}}|^2 d\omega \stackrel{\substack{\text{Module of the complex} \\ \text{exponential}}}{=} \frac{1}{2\pi} \int_{-\pi}^{\pi} 1^2 d\omega = \frac{1}{2\pi} \omega|_{-\pi}^{\pi} = \frac{1}{2\pi} [\pi - (-\pi)] = 1$$

In the expression, we have a constant term and another one that depends on the coefficients of the  $N$ -th-order FIR filter. Truncating the summation indices we get

$$\approx 1 + \sum_{n=0}^N [h^2(n) - 2h(n) \operatorname{sinc}(n - \mathcal{D})]$$

The optimal solution for a  $N$ -th order filter in the  $L_2$  sense is the one with  $N + 1$  coefficients truncated symmetrically around the peak of  $h_{id}(n)$ . Thus, the impulse response of the LS fractional delay FIR filter corresponds to the truncation of the ideal response. The  $\operatorname{sinc}()$  function is in fact truncated at zero to be causal and limited to an order  $N$ .

$$h(n) = \begin{cases} \operatorname{sinc}(n - \mathcal{D}), & \text{for } 0 \leq n \leq N \\ 0, & \text{otherwise} \end{cases}$$

For a given  $\mathcal{D}$ , the optimal filter order satisfies

$$|\mathcal{D}| - \frac{N}{2} = 0, \quad \text{for } N \text{ even}$$

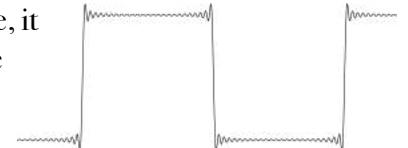
$$|\mathcal{D}| - \frac{N-1}{2} = 0, \quad \text{for } N \text{ odd}$$

At the end we can write the resulting approximation error as

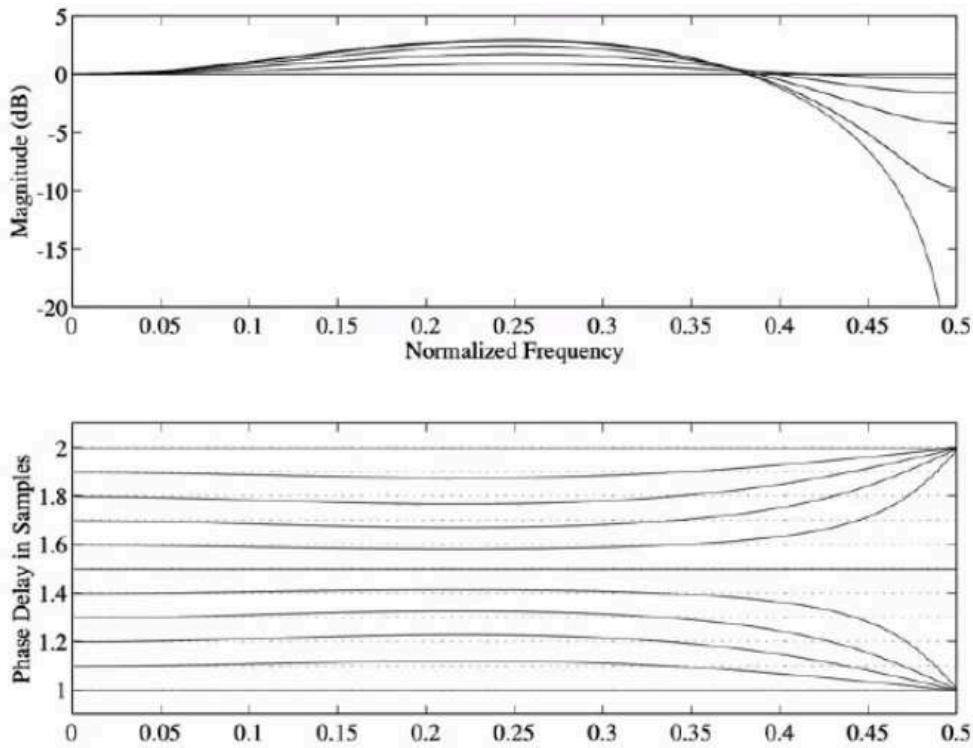
$$\|E(\omega)\|_2^2 = \sum_{n=-\infty}^{-1} |h_{id}(n)|^2 + \sum_{n=N+1}^{\infty} |h_{id}(n)|^2$$

The solution is surely very intuitive but despite being optimal in the  $L_2$  sense, it suffers from Gibbs phenomenon: in reality it is impossible to have an infinite order FIR filter to approximate the changing of the slope, therefore we get some undesired ripple in the magnitude response close to  $\omega = \pi$ .

Furthermore, positive gains (in dB) at some frequencies become problematic in systems with feedback.



As we can notice, their approximation is not that much perfect. For some values of delay, the magnitude is very distorted and also the phase is affected by some modifications.



### **Windowing methods**

How do we alleviate this kind of phenomena? A method for reducing Gibbs phenomena is to use a window function  $w(n)$  different from rectangular.

$$h(n) = \begin{cases} w(n - D) \operatorname{sinc}(n - D), & \text{for } 0 \leq n \leq N \\ 0, & \text{otherwise} \end{cases}$$

The midpoint of the window function  $w(n)$  of length  $(N + 1)$  is shifted by  $D$  so that the shifted sinc function is windowed symmetrically with respect to its centre. Many window functions can be delayed by fractional values (e.g. Hamming). Larger  $L_2$  error norm than truncated LS solution. Unfortunately, it is difficult to control magnitude error by adjusting window parameters.

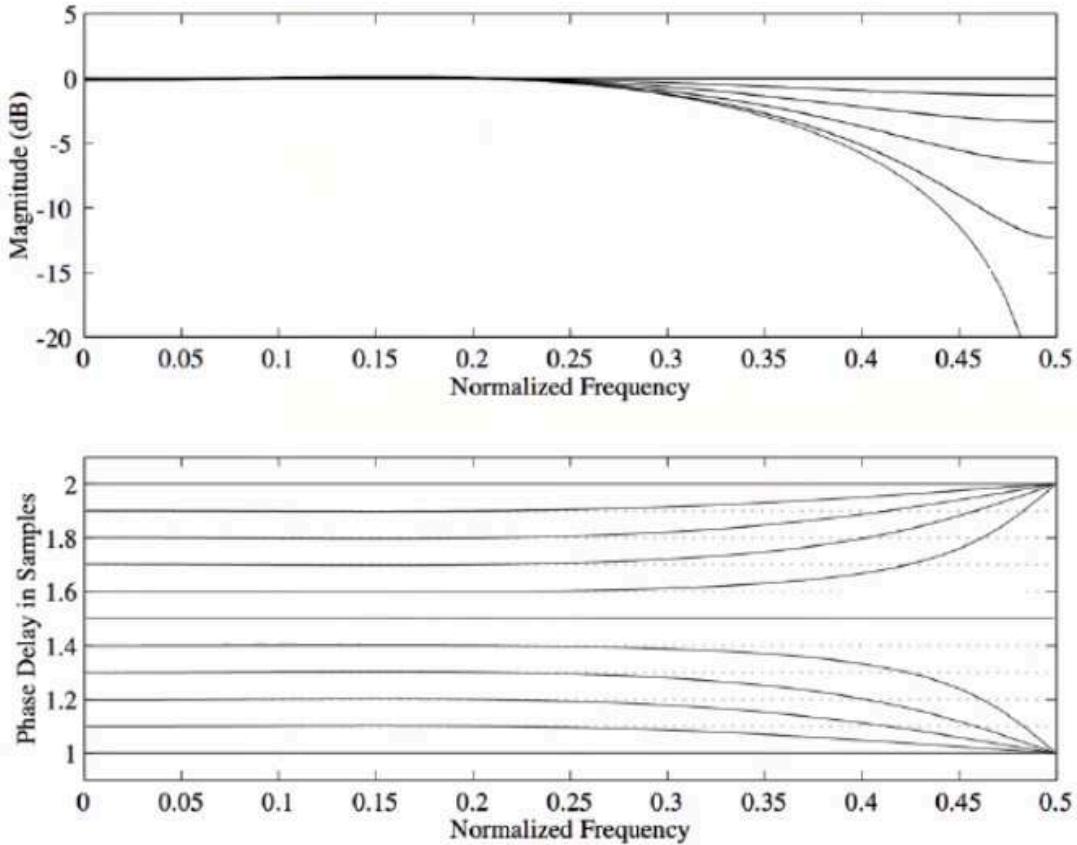
### **GLS Methods**

Another approach that can be used to overshoot the Gibbs phenomena is the GLS (General Least Square) FIR approximation. Instead of windowing the impulse response of FIR fractional delay filter, we could window the  $L_2$  norm of the error in the frequency domain.

$$\|E(\omega)\|_2^2 = \frac{1}{2\pi} \int_{-\alpha\pi}^{\alpha\pi} W(\omega) |E(\omega)|^2 d\omega = \frac{1}{2\pi} \int_{-\alpha\pi}^{\alpha\pi} W(\omega) |H(\omega) - H_{id}(\omega)|^2 d\omega$$

Where the error is defined in the lowpass frequency band  $[-\alpha\pi, \alpha\pi]$ . We are just imposing our error function to be minimal.

And here there's the magnitude response of our filter.



### Lagrange Interpolation

Another approach we can use is the Lagrange Interpolation. This method simply makes the error function maximally flat around  $\omega_0 = 0$  by setting derivatives to 0.

$$\left. \frac{d^l E(\omega)}{d\omega^l} \right|_{\omega=\omega_0} = 0 \quad \text{for } l = 0, \dots, N$$

Which can be rewritten as

$$\begin{aligned} \left. \frac{d^l E(\omega)}{d\omega^l} \right|_{\substack{\omega=\omega_0=0 \\ E(\omega)=H(\omega)-H_{id}(\omega)}} &= \left. \frac{d^l [H(\omega) - H_{id}(\omega)]}{d\omega^l} \right|_{\substack{\omega=\omega_0=0 \\ H_{id}(\omega) \text{ shows a maximum} \\ \text{in correspondence of } \omega=\omega_0}} \\ &\quad \text{Therefore,} \\ &\quad \left. \frac{d^l H_{id}(\omega)}{d\omega^l} \right|_{\omega=\omega_0} = 0 \quad \forall l \\ &= \left. \frac{d^l H(\omega)}{d\omega^l} \right|_{\substack{\omega=0 \\ H(\omega)=e^{-j\omega D}}} \stackrel{\omega}{=} \left. \frac{d^l (e^{-j\omega D})}{d\omega^l} \right|_{\omega=0} \end{aligned}$$

To solve these derivatives, we switch back to discrete time domain

$$\left. \frac{d^l H(\omega)}{d\omega^l} \right|_{\substack{\omega=0 \\ "DTFT"\{H(\omega)\}=\sum_{n=0}^N h(n)e^{-j\omega n}}} = \left. \frac{d^l \sum_{n=0}^N h(n) e^{-j\omega n}}{d\omega^l} \right|_{\omega=0}$$

What we can do is computing the derivative for the different  $l$ .

- $l = 0$

$$\left. \frac{d^0 \sum_{n=0}^N h(n) e^{-j\omega n}}{d\omega^0} \right|_{\omega=0} = \sum_{n=0}^N h(n) e^{-j\omega n} \Big|_{\omega=0} = \sum_{n=0}^N h(n) \stackrel{\substack{\text{Condition} \\ \text{found} \\ \text{before}}}{=} \left. \frac{d^0(e^{-j\omega D})}{d\omega^0} \right|_{\omega=0} = 1$$

The condition we have to impose when  $l = 0$  is that the sum of all the filter taps is equal to one.

$$\sum_{n=0}^N h(n) = 1$$

- $l = 1$

$$\begin{aligned} \left. \frac{d^1 \sum_{n=0}^N h(n) e^{-j\omega n}}{d\omega^1} \right|_{\omega=0} &= \sum_{n=0}^N h(n) \left. \frac{de^{-j\omega n}}{d\omega} \right|_{\omega=0} = \sum_{n=0}^N h(n) (-j)n e^{-j\omega n} \Big|_{\omega=0} = \\ &= -j \sum_{n=0}^N nh(n) \stackrel{\substack{\text{Condition} \\ \text{found} \\ \text{before}}}{=} \left. \frac{d^1(e^{-j\omega D})}{d\omega^1} \right|_{\omega=0} = -jD e^{-j\omega D} \Big|_{\omega=0} = -jD \end{aligned}$$

The condition we have to impose when  $l = 1$  is that the sum of all the filter taps is equal to one.

$$-j \sum_{n=0}^N nh(n) = -jD$$

So that

$$\sum_{n=0}^N nh(n) = D$$

- $l = 2$

$$\begin{aligned} \left. \frac{d^2 \sum_{n=0}^N h(n) e^{-j\omega n}}{d\omega^2} \right|_{\omega=0} &= \sum_{n=0}^N h(n) \left. \frac{d^2 e^{-j\omega n}}{d\omega^2} \right|_{\omega=0} = \sum_{n=0}^N h(n) (-j)^2 n^2 e^{-j\omega n} \Big|_{\omega=0} = \\ &= \sum_{n=0}^N n^2 h(n) \stackrel{\substack{\text{Condition} \\ \text{found} \\ \text{before}}}{=} \left. \frac{d^2(e^{-j\omega D})}{d\omega^2} \right|_{\omega=0} = (-j)^2 D^2 e^{-j\omega D} \Big|_{\omega=0} = D^2 \end{aligned}$$

The condition we have to impose when  $l = 2$  is that the sum of all the filter taps is equal to one.

$$\sum_{n=0}^N n^2 h(n) = D^2$$

- Any  $l$

From these firsts iterations we infer that, for any  $l$

$$\left. \frac{d^l \sum_{n=0}^N h(n) e^{-j\omega n}}{d\omega^l} \right|_{\omega=0} = \sum_{n=0}^N n^l h(n) \stackrel{\substack{\text{Condition} \\ \text{found} \\ \text{before}}}{=} \left. \frac{d^l (e^{-j\omega D})}{d\omega^l} \right|_{\omega=0} = \mathcal{D}^l$$

Therefore

$$\sum_{n=0}^N n^l h(n) = \mathcal{D}^l$$

Thus, the problem

$$\left. \frac{d^l E(\omega)}{d\omega^l} \right|_{\omega=\omega_0} = 0 \quad \text{for } l = 0, \dots, N$$

becomes

$$\sum_{n=0}^N n^l h(n) = \mathcal{D}^l \quad \text{with } l = 0, \dots, N$$

That represents a set of  $N + 1$  linear equations. Collecting all of them in a matrix form we can we write the problem in matrix fashion

$$\underline{\underline{V}} \underline{h} = \underline{v}$$

Where:

- $\underline{\underline{V}}$  is a Vandermonde matrix  $(N + 1) \times (N + 1)$

$$\underline{\underline{V}} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 0 & 1 & 2 & \dots & N \\ 0 & 1 & 2^2 & \dots & N^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & 2^N & \dots & N^N \end{pmatrix}$$

- $\underline{h} = [h(1), h(2), \dots, h(N)]^T$  is the set of coefficients to be found
- $\underline{v} = [1, D, D^2, \dots, D^N]^T$

**Reminder: Vandermonde Matrix**

A **Vandermonde matrix** is a matrix with the terms of a geometric progression in each row: an  $m + 1 \times (n + 1)$  matrix

$$\underline{\underline{V}} = \underline{\underline{V}}(x_0, x_1, x_2, \dots, x_m) = \begin{pmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \dots & x_m^n \end{pmatrix}$$

with entries  $\underline{\underline{V}}_{i,j} = x_i^j$ , the  $j^{th}$  power of the number  $x_i$ , for all zero-based indices  $i$  and  $j$ .

The determinant of a square Vandermonde matrix (when  $n = m$ ) is called a Vandermonde determinant or Vandermonde polynomial. Its value is:

$$\det(\underline{\underline{V}}) = \prod_{0 \leq i < j \leq n} (x_j - x_i)$$

This is non-zero if and only if all  $x_i$  are distinct (no two are equal), making the Vandermonde matrix invertible.

The Vandermonde matrix has a lot of interesting properties. As it is a singular matrix, a solution to

$$\underline{\underline{V}} \underline{h} = \underline{\underline{v}}$$

exists in closed form. The “maximally flat” fractional delay filter is found by

$$\underline{h} = \underline{\underline{V}}^{-1} \underline{\underline{v}}$$

The solution can be conveniently rewritten as

$$h(n) = \prod_{\substack{k=0, \\ k \neq n}}^N \frac{D - k}{n - k}, \quad n = 0, \dots, N$$

These coefficients are equal to those of the Lagrange interpolation formula for equally spaced abscissas. Again we can approximate optimal order for our filters. Optimal orders  $N$  can be found for a given delay  $D$ , but we need to set

$$N > |\mathcal{D}|$$

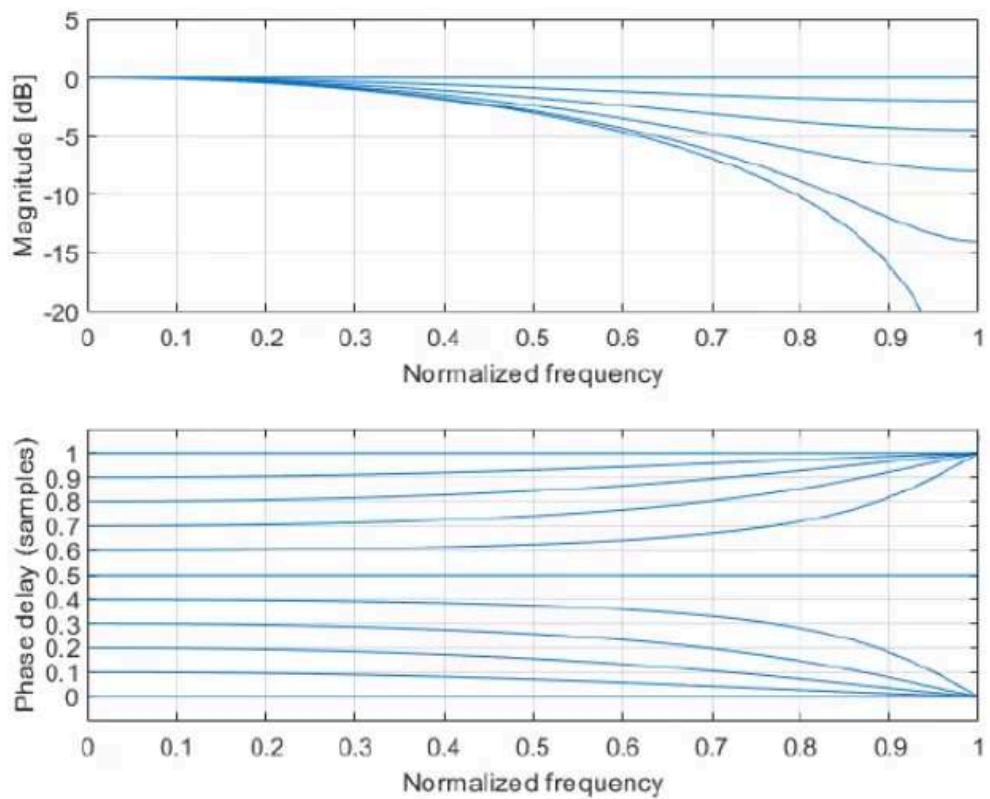
We notice that by setting  $N = 1$ , we perform a simple linear interpolation

$$h(0) = 1 - D, \quad h(1) = D$$

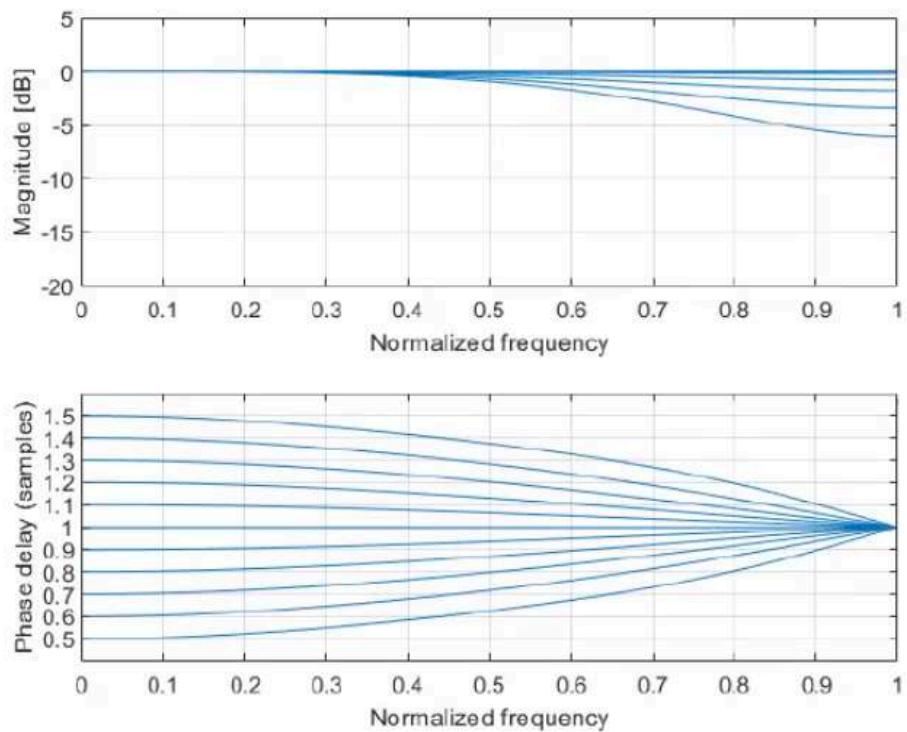
In general:

- $N$  odd: exact linear phase but magnitude response has a zero at  $\omega = \pi$ ;
- $N$  even: better magnitude response but worse phase response.

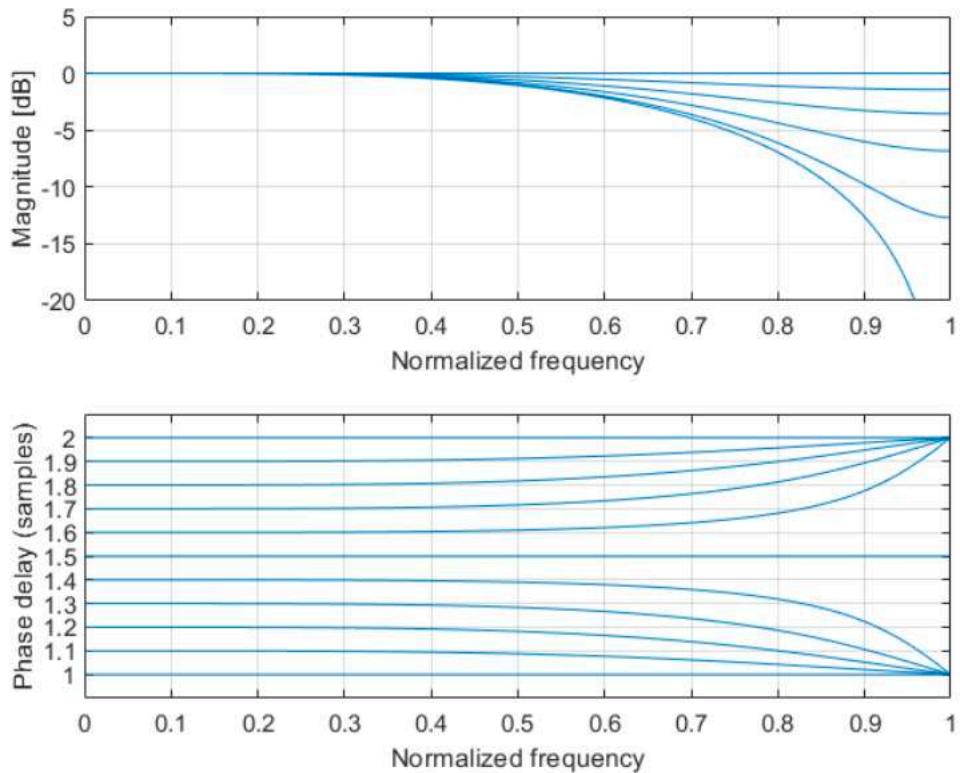
Let us see how they appear when plotted:



For second order Lagrange interpolator we have a very good magnitude but a messy phase



And with an order equal to 3, the phase is almost linear but the magnitude is not flat.



Let us compare the different results obtained with different techniques on Matlab

```

clear
close all
clc

% parameters
fs = 44100; % sampling frequency [Hz]
tau = 0.0003; % delay [s]

D = tau*fs

N = 2*floor(D);
n = 0:N;

%% Least squares approximation
h_ls = sinc(n-D);

figure
stem(n, h_ls), xlabel('Samples'), ylabel('Impulse response')
title('Least Squares Approximation')

[H_ls, f] = freqz(h_ls, 1, 'half', fs);

figure,
subplot(2,1,1)
plot(f*fs/pi/2, abs(H_ls)), xlabel('Frequency [Hz]'), ylabel('Magnitude [dB]')
axis([0, fs/2, 0 2]), grid
title('Least Squares Approximation')
subplot(2,1,2)
grpdelay(h_ls, 1, 1024, 'half', fs)

%% Windowing
h_win = h_ls .* hamming(N+1);

figure
stem(n, h_win), xlabel('Samples'), ylabel('Impulse response')

```

```

title('Windowing of LS Approximation')

[H_win, f] = freqz(h_win, 1, 'half', fs);

figure,
subplot(2,1,1)
plot(f*fs/pi/2, abs(H_win)), xlabel('Frequency [Hz]'), ylabel('Magnitude [dB]')
axis([0, fs/2, 0 2]), grid
title('Windowing of LS Approximation')
subplot(2,1,2)
grpdelay(h_win, 1, 1024, 'half', fs)

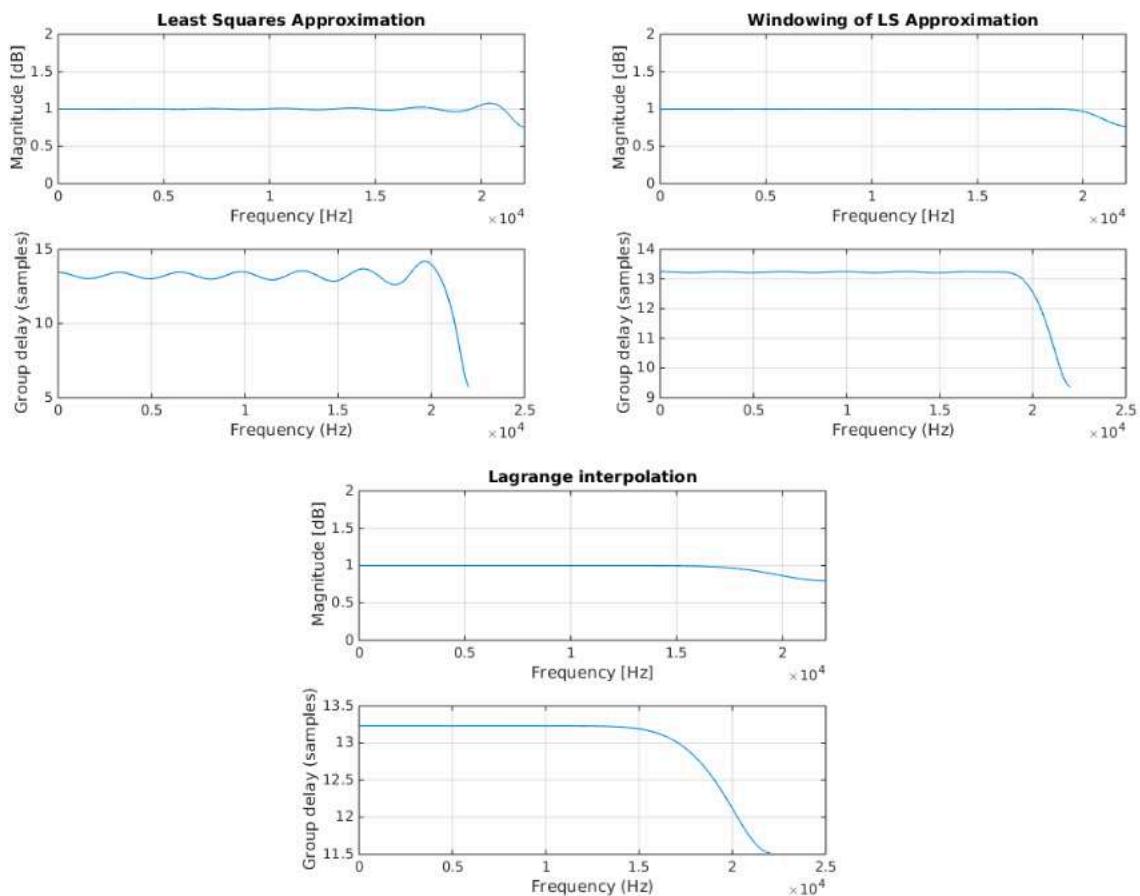
%% Lagrange interpolation
h_int = zeros(N+1,1);
for nn = 0:N
    k = (0:N)';
    k = k(k~=nn);
    h_int(nn+1) = prod((D-k)./(nn-k));
end

figure
stem(n, h_int), xlabel('Samples'), ylabel('Impulse response')
title('Lagrange interpolation')

[H_int, f] = freqz(h_int, 1, 'half', fs);

figure,
subplot(2,1,1)
plot(f*fs/pi/2, abs(H_int)), xlabel('Frequency [Hz]'), ylabel('Magnitude [dB]')
axis([0, fs/2, 0 2]), grid
title('Lagrange interpolation')
subplot(2,1,2)
grpdelay(h_int, 1, 1024, 'half', fs)

```



This concludes the first part related to fractional delays.

## Integer and fractional delay filters – IIR Filters

We have seen how to design FIR filters to approximate fractional delays. Now we show how IIR filters can be used for the same design problem. IIR filters meet same frequency domain specification with minor order than FIR filters in the face of more complex design. Furthermore, they can possibly be unstable (we remind that all the poles must remain with unit circle in the z-domain to assure stability) and it can leads to difficult real time update.

All-pass filters are typically used for phase equalization and other signal processing tasks where phase characteristics are of great importance. In order to implement it, we want to obtain a function that has

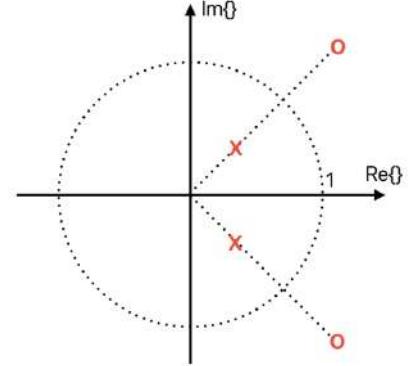
$$|H_{id}(\omega)| = |e^{-j\omega D}| = 1 \quad \angle H_{id}(\omega) = -\omega D$$

A response like this one can be optioned by an all-pass filter written as

$$H_{AP}(z) = \frac{z^{-N} A(z^{-1})}{A(z)} = \frac{a_N + a_{N-1}z^{-1} + \dots + a_1z^{-(N-1)} + z^{-N}}{1 + a_1z^{-1} + \dots + a_{N-1}z^{-(N-1)} + a_Nz^{-N}}$$

Description of the function:

- The numerator polynomial is a mirrored version of the denominator polynomial.
- Zeros at the numerator have corresponding poles at the denominator.
- Poles lie inside the unit circle in the complex plane (stability), zeros lie outside.
- Thus, the magnitude response of an all-pass filter is flat.



If we look at this function in the complex plane, we notice that there is always a zero symmetrical to the pole. Furthermore, all the poles are inside the conference to ensure stability.

Let us point out our problem:

We have a  $N$ -th order allpass filter

$$H_{AP}(z) = \frac{a_N + a_{N-1}z^{-1} + \dots + a_1z^{-(N-1)} + z^{-N}}{1 + a_1z^{-1} + \dots + a_{N-1}z^{-(N-1)} + a_Nz^{-N}}$$

That, by applying the formal substitution  $z = e^{j\omega}$ , has:

Module equal to:

Phase equal to:  
 $\angle H_{AP}(\omega)$

$$|H_{AP}(e^{j\omega})| = \left| e^{-j\omega N} \frac{A(e^{-j\omega})}{A(e^{j\omega})} \right| \equiv 1$$

Where:

$$H_{AP}(e^{j\omega}) = e^{j\angle H_{AP}(\omega)}$$

Our objective is to match the phase  $\angle H_{AP}(\omega)$  with that of the ideal fractional delay filter. How can we do so? Thanks to the property for which the phase of a ratio can be written as the difference between the phase of the numerator and the one of the denominator, we rewrite the phase as it follows

$$\angle H_{AP}(\omega) = \angle \left\{ e^{-j\omega N} \frac{A(e^{-j\omega})}{A(e^{j\omega})} \right\} = \angle \{ e^{-j\omega N} A(e^{-j\omega}) \} - \angle A(e^{j\omega}) =$$

$$\begin{aligned}
&= \angle\{e^{-j\omega N}\} - \angle\{A(e^{j\omega})\} - \angle A(e^{j\omega}) = -\omega N + \angle\left\{\frac{1}{A(e^{j\omega})}\right\} + \angle\left\{\frac{1}{A(e^{j\omega})}\right\} = \\
&= -N\omega + 2\angle\left\{\frac{1}{A(e^{j\omega})}\right\}
\end{aligned}$$

Where  $\angle\left\{\frac{1}{A(e^{j\omega})}\right\}$  is the phase response of all-pole filter. Therefore, if we know how to design the phase response of an all-pole filter (Autoregressive filter, AR), we also know how to design the corresponding all-pass filter using what we have just write above.

So, the problem becomes how to evaluate the phase response of the all-pole filter  $\frac{1}{A(e^{j\omega})}$ .

We can rewrite it as

$$\begin{aligned}
&\angle\left\{\frac{1}{A(e^{j\omega})}\right\} = \arg\left\{\frac{1}{A(e^{j\omega})}\right\} = \\
&= \arg\{A(e^{-j\omega})\} \stackrel{\text{Definition of phase for a complex number}}{=} \\
&\quad A(e^{-j\omega}) = 1 + a_1 e^{-j\omega} + a_2 e^{-j2\omega} + \dots + a_N e^{-jN\omega} = \\
&\stackrel{\text{Euler}}{=} 1 + a_1(\cos(-\omega) + j \sin(-\omega)) + a_2(\cos(-\omega) + j \sin(-2\omega)) + \dots + a_N(\cos(-N\omega) + j \sin(-N\omega)) \\
&\angle A(e^{-j\omega}) \triangleq \arctan\left\{\frac{\Im\{A(e^{-j\omega})\}}{\Re\{A(e^{-j\omega})\}}\right\} = \arctan\left\{\frac{\sum_{n=0}^N a_n \sin(n\omega)}{\sum_{n=0}^N a_n \cos(n\omega)}\right\} \\
&= \arctan\left\{\frac{\sum_{n=0}^N a_n \sin(n\omega)}{\sum_{n=0}^N a_n \cos(n\omega)}\right\} = \arctan\left\{\frac{\underline{a}^T \underline{s}}{\underline{a}^T \underline{c}}\right\}
\end{aligned}$$

Where:

$$\underline{a} = \begin{pmatrix} 1 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{pmatrix}, \quad \underline{s} = \begin{pmatrix} 1 \\ \sin(\omega) \\ \sin(2\omega) \\ \vdots \\ \sin(N\omega) \end{pmatrix}, \quad \underline{c} = \begin{pmatrix} 1 \\ \cos(\omega) \\ \cos(2\omega) \\ \vdots \\ \cos(N\omega) \end{pmatrix}$$

Therefore, the problem of before, can be re-written in this terms

$$\angle H_{AP}(\omega) = -N\omega + 2 \arctan\left\{\frac{\underline{a}^T \underline{s}}{\underline{a}^T \underline{c}}\right\}$$

Whose corresponding group delay is

$$\tau_{g,AP(\omega)} = -\frac{d\angle H_{AP}(\omega)}{\omega} = N - 2\tau_{g,A(\omega)}$$

From which we can derive the phase delay as

$$\tau_{g,AP(\omega)} = -\frac{\angle H_{AP}(\omega)}{\omega} = N - 2\tau_{p,A(\omega)}$$

Reminder: group and phase delay

Group delay and Phase delay are two related ways of describing how a signal's frequency components are delayed in time when passing through a linear time-invariant (LTI) system (such as a microphone, coaxial cable, amplifier, loudspeaker, telecommunications system, ethernet cable, digital filter, or analogue filter). Phase delay describes the time shift of a sinusoidal component (a sine wave in steady state). Group delay describes the time shift of the envelope of a wave packet, a "pack" or "group" of oscillations centred around one frequency that travel together, formed for instance by multiplying (amplitude modulation) a sine wave by an envelope (such as a tapering function).

Just by looking at this result, we notice that there is a great difference between the design of FIR and all-pass filters:

- The coefficients of an FIR filter are easily obtained as the inverse DTFT of the frequency domain specifications;
- The relationship between the coefficients of the all-pass transfer function and the samples of the corresponding impulse response is not that simple;
- Most design techniques for all-pass filters are iterative.

The desired or ideal phase response in the fractional delay approximation problem is:

$$\angle H_{id}(\omega) = -\omega D$$

At this point, we can define a phase error in order to solve the problem. We define it as the deviation from the desired phase function:

$$\Delta\Theta = \angle H_{id} - \angle H_{AP} \underset{\omega}{=} \angle H_{id} + N\omega - 2 \arctan \left\{ \frac{\underline{a}^T \underline{s}}{\underline{a}^T \underline{c}} \right\} = 2 \arctan \left\{ \frac{\underline{a}^T \underline{s}_\beta}{\underline{a}^T \underline{c}_\beta} \right\}$$

$$\angle H_{AP}(\omega) = -N\omega + 2 \arctan \left\{ \frac{\underline{a}^T \underline{s}}{\underline{a}^T \underline{c}} \right\}$$

Where:

$$\underline{a} = \begin{pmatrix} 1 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{pmatrix}, \quad \underline{s} = \begin{pmatrix} 1 \\ \sin(\beta(\omega)) \\ \sin(\beta(\omega) - \omega) \\ \vdots \\ \sin(\beta(\omega) - N\omega) \end{pmatrix}, \quad \underline{c} = \begin{pmatrix} 1 \\ \cos(\beta(\omega)) \\ \cos(\beta(\omega) - \omega) \\ \vdots \\ \cos(\beta(\omega) - N\omega) \end{pmatrix}$$

$$\text{and } \beta(\omega) = \frac{1}{2} [\angle H_{id}(\omega) + N\omega].$$

By approximating a fractional delay as a integer part and a fractional one  $D = N + d$ , the term  $\beta$  reduces to

$$\beta(\omega) = \frac{1}{2} [\angle H_{id}(\omega) + N\omega] \underset{D=N+d \rightarrow N=D-d}{=} \frac{1}{2} [\angle H_{id}(\omega) + D\omega - \omega d] \underset{\angle H_{id}(\omega) = -\omega D}{=} -\frac{\omega d}{2}$$

So, the approach is the same that we ever followed: we minimise the weighted least-squares phase error.

$$\|E_\Theta\|_2^2 = \frac{1}{2\pi} \int_{-\pi}^{\pi} W(\omega) |\Delta\Theta(\omega)|^2 d\omega$$

Where  $W(\omega)$  is a non-negative weighting function.

An alternative definition is the least-square *phase delay error*

$$\|E_p\|_2^2 = \frac{1}{2\pi} \int_{-\pi}^{\pi} W(\omega) \left| \frac{\Delta\Theta(\omega)}{\omega} \right|^2 d\omega = \frac{1}{2\pi} \int_{-\pi}^{\pi} \frac{W(\omega)}{\omega^2} |\Delta\Theta(\omega)|^2 d\omega$$

There are many algorithms that can design all pass filters with desired phase, group delay, or phase delay (for instance we can use equiripple to design a filter with a certain ripple; minimax, etc...). A well-suited digital AP (allpass) filter for FD (fractional delay) approximation in audio signal processing is the maximally flat design at  $\omega = 0$ . It can be performed by following the Thiran approach.

### Thiran all-pole LP design

Jean-Philippe Thiran (born in 1970) proposed an analytic solution to obtain the coefficients of an all-pole low-pass filter with a maximally flat group delay response at the zero frequency: a closed form solution known as the **maximally-flat group-delay design**. For an all-pole low-pass filter with transfer function  $\frac{1}{A(z^{-1})}$ , the condition of maximally flat group delay at  $\omega_0 = 0$  yields

$$a_n = (-1)^n \binom{N}{n} \prod_{i=0}^N \frac{2\mathcal{D} + l}{2\mathcal{D} + n + l} \quad \mathcal{D} > 0$$

For positive delays, the poles lie in the unit circle in the complex plane (stable filter). Unfortunately, there is a drawback: the magnitude response of the all-pole lowpass filter cannot be controlled. However, we can turn this design into an all pass design, which has unitary gain at all frequencies. Since we know that the group delay of this kind of filter is closely related to that of an allpass filter, we can exploit this kind of estimation to find the coefficients of the all pass filter. What we get is the following expression:

$$a_n = (-1)^n \binom{N}{n} \prod_{i=0}^N \frac{\mathcal{D} + l}{\mathcal{D} + n + l} \quad l = 0, 1, 2, \dots, N$$

We notice that it is very similar to the previous one: the only difference lies in the fact that there is no more factor two before  $\mathcal{D}$ .

Let us calculate some coefficients for different value of  $N$ .

	$a_1$	$a_2$	$a_3$
$N = 1$	$-\frac{\mathcal{D} - 1}{\mathcal{D} + 1}$		
$N = 2$	$-2 \frac{\mathcal{D} - 2}{\mathcal{D} + 1}$	$\frac{(\mathcal{D} - 1)(\mathcal{D} - 2)}{(\mathcal{D} + 1)(\mathcal{D} + 2)}$	
$N = 3$	$-3 \frac{\mathcal{D} - 3}{\mathcal{D} + 1}$	$3 \frac{(\mathcal{D} - 2)(\mathcal{D} - 3)}{(\mathcal{D} + 1)(\mathcal{D} + 2)}$	$-\frac{(\mathcal{D} - 1)(\mathcal{D} - 2)(\mathcal{D} - 3)}{(\mathcal{D} + 1)(\mathcal{D} + 2)(\mathcal{D} + 3)}$

Let us do some considerations. The first-order all-pass filter transfer function is

$$H(z) = \frac{a_1 + z^{-1}}{1 + a_1 z^{-1}} \quad a_1 < 1$$

Its phase response can be approximated at lower frequencies as it follows

$$\begin{aligned} \arg\{H(\omega)\} &= \arg\{H(z)|_{z=e^{j\omega_0}}\} = \arg\left\{\frac{a_1 + e^{-j\omega_0}}{1 + a_1 e^{-j\omega_0}}\right\}_{Euler} = \\ &= \arg\left\{\frac{a_1 + \cos(-\omega_0) + j \sin(-\omega_0)}{1 + a_1(\cos(-\omega_0) + j \sin(-\omega_0))}\right\} = \arg\left\{\frac{a_1 + \cos(\omega_0) - j \sin(\omega_0)}{1 + a_1 \cos(\omega_0) - ja_1 \sin(\omega_0)}\right\} = \\ &= \arg\{a_1 + \cos(\omega_0) - j \sin(\omega_0)\} - \arg\{1 + a_1 \cos(\omega_0) - ja_1 \sin(\omega_0)\} = \\ &= \arctan\left(-\frac{\sin(\omega_0)}{a_1 + \cos(\omega_0)}\right) - \arctan\left(-\frac{a_1 \sin(\omega_0)}{1 + a_1 \cos(\omega_0)}\right)_{\arctan(-\theta) = -\arctan(\theta)} = \\ &= -\arctan\left(\frac{\sin(\omega_0)}{a_1 + \cos(\omega_0)}\right) + \arctan\left(\frac{a_1 \sin(\omega_0)}{1 + a_1 \cos(\omega_0)}\right)_{\substack{\text{small } \omega_0 \\ \arctan(\alpha) \approx \alpha \\ \text{small } \alpha}} \approx \\ &\approx -\frac{\sin(\omega_0)}{a_1 + \cos(\omega_0)} + \frac{a_1 \sin(\omega_0)}{1 + a_1 \cos(\omega_0)}_{\substack{\text{small } \omega_0 \\ \cos(\omega_0) \approx 1 \\ \sin(\omega_0) \approx \omega_0}} \approx -\frac{\omega_0}{a_1 + 1} + \frac{a_1 \omega_0}{1 + a_1} = -\omega_0 \frac{1 - a_1}{1 + a_1} \end{aligned}$$

Therefore, the phase response approximately linear with phase and group delay approximately equal to  $\frac{1-a_1}{1+a_1}$  where  $a_1$  is in turn equal to  $a_1 = \frac{1-\mathcal{D}}{1+\mathcal{D}}$ . This is graphically, what we get:

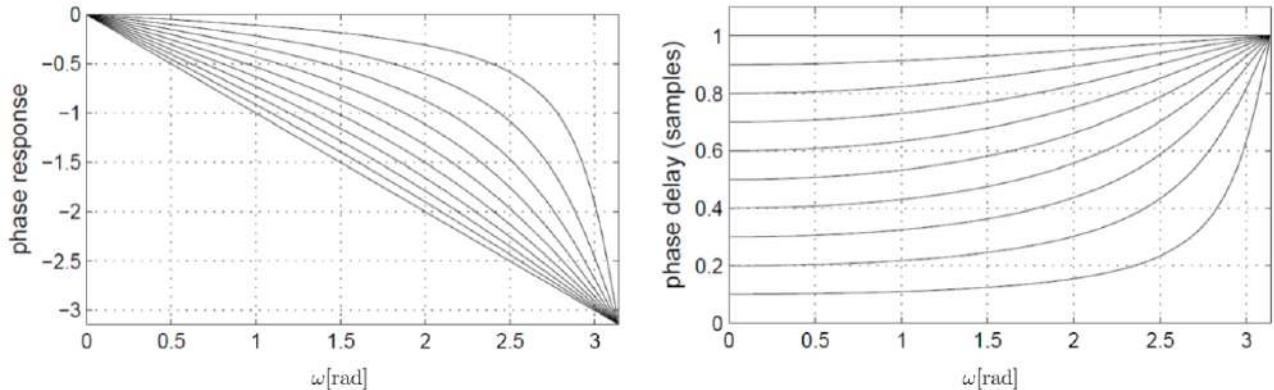


Figure: Phase response and phase delay of a first-order Thiran allpass filter.

As we can notice, the phase is not perfectly linear. By increasing the order of the filter, also the phase increases its linearity.

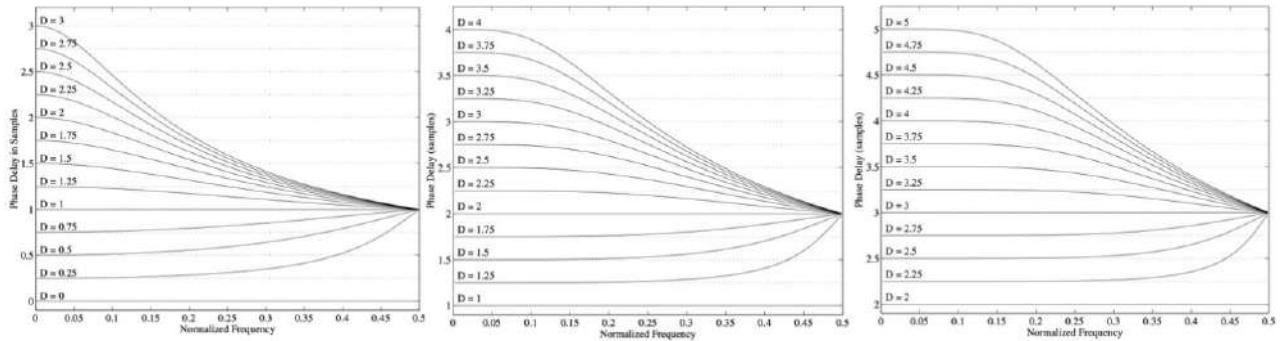


Figure: Phase delay of a first-order (left), second-order (middle) and third-order (right) Thiran allpass filter. The dashed lines indicate the ideal phase delay, which is constant.

If we want to define the Thiran filter on Matlab we can use the following function:

*To initialise the line*

```
function f = thirandline_init(D,N) %uses Nth order Thiran filter
    f.x = 0; f.y = 0;
    f.D = D;
    f.in = zeros(1, floor(D)-N); %set delay (not necessarily integer)
    f.state = zeros(1,N); %create buffer for past input values
    f.a = zeros(1, N+1); %coefficients of the Thiran filter
    d = D-floor(D); %fractional delay to approximate
    for k = 0 : N
        ak = 1;
        for l = 0 : N
            ak = ak * (d+1)/(d+k+l);
        end
        f.a(k+1) = (-1)^k * nchoosek(N,k) * ak;
    end
end
```

*Function to update the state*

```
function f = thirandline_compute(f)
    [out, state] = filter(fliplr(f.a), f.a, f.in(1), f.state);
    f.state = state;
    f.y = out;
    f.in = [f.in(2:length(f.in)), f.x]; %update buffer
end
```

## References

- The discussion presented in this section follows some excerpts of [4].
- In [4] more techniques for fractional delay approximation with FIR and IIR filters are presented.
- Industry standard delay filters are discussed in [5].
- A detailed implementation of a fractionally-addressed delay structure can be found in [6].
- Design according to error criterion alternative to L<sub>2</sub> can be found in [7].

## Delay-based effects - Comb filters

Analysing the effects we can obtain by using delay filters, we will work with integer delays for simplicity. Obviously, we could use also fractional delay filters. Anyway, by considering an integer delay, the response of the filter is described by the following different equation (**FIR**)

$$y(n) = x(n) + gx(n - M) \quad \text{with } M = \lfloor \tau f_s \rfloor$$

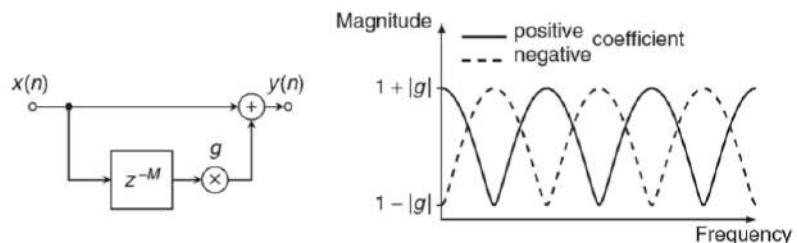
Where:

- $\tau$  is the amount of time delay (in seconds)
- $g$  is a gain factor for the delayed signal

The transfer function of a filter like this one is:

$$H(z) = 1 + g z^{-M}$$

On the right is reported the filter structure and magnitude response.



Depending on the gain  $g$ , the filter can either amplify or attenuate the signal put as input:

- For  $g > 0$  the filter amplifies all frequencies multiple to  $\frac{1}{\tau}$  and attenuates frequencies in between;
- For  $g < 0$  the filter attenuates all frequencies multiple to  $\frac{1}{\tau}$  and amplifies frequencies in between.

From time domain point of view, what we can say is that:

- if  $\tau$  is large we ear a separate replica of the signal and we can barely identify spectral effects;
- if  $\tau$  is small we can't ear separate events but spectral effects are noticeable.

Here again, an example on MATLAB with  $\tau = 1 \text{ ms}$ ,  $g = 0.5$

```
%% FIR comb filter
% Lucio Bianchi - 03/04/2015

clear
close all
clc

fs = 44100;

% unit impulse
N = 100;
n = 0:N-1;
x = zeros(N,1);
x(1) = 1;

% delay line
tau = 0.001; % delay [seconds]
M = floor(tau*fs);
delayline = zeros(M, 1);

% delay gain
g = 0.5;

y = zeros(N, 1);
for ii = 1:N
    y(ii) = x(ii) + g * delayline(ii);
end
```

```

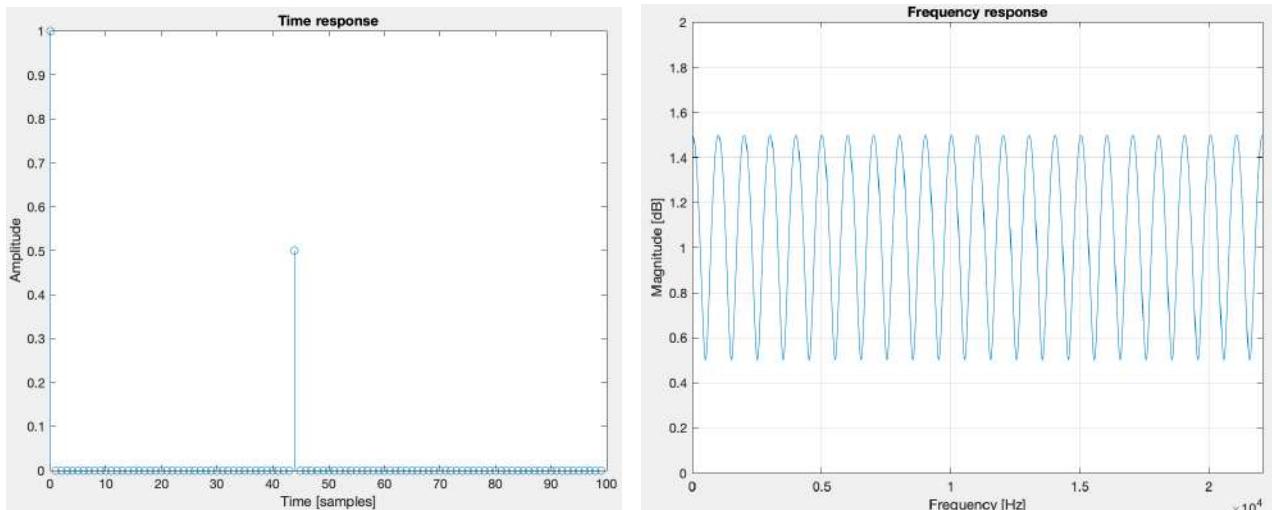
y(ii) = x(ii) + g*delayline(end);
delayline = [x(ii); delayline(1:end-1)];
end

figure,
stem(n,y), xlabel('Time [samples]'), ylabel('Amplitude')
title('Time response')

[H, f] = freqz(y, 1, 'half', fs);

figure,
plot(f*fs/pi/2, abs(H)), xlabel('Frequency [Hz]'), ylabel('Magnitude [dB]')
axis([0, fs/2, 0 2]), grid
title('Frequency response')

```

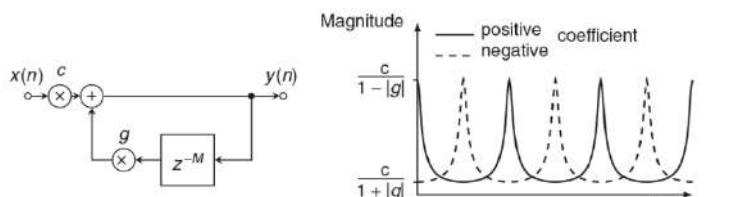


We could implement the delay line also with a **IIR** filter. The difference equation would be:

$$y(n) = cx(n) + gy(n - M) \quad \text{with } M = \lfloor \tau f_s \rfloor$$

And the transfer function:

$$H(z) = \frac{c}{1 - gz^{-M}}$$



On the right is reported again the filter structure and magnitude response.

Considerations:

- Scale factor  $c$  is necessary since this structure gives high amplification
- Infinite time response
- Every  $\tau$  a copy of the input signal is produced with amplitude  $g^p$  ( $p$  is the number of times the signal passed through the delay line)
- The structure is stable if  $|g| \leq 1$

This case again, let us see how to implement the filter on Matlab

```
%% IIR comb filter
% Lucio Bianchi - 04/04/2015

% Remember there is an error in Figure 3.2 by Zolzer (+ and - should be
switched)

clear
close all
clc

fs = 44100;

% unit impulse
N = 100;
n = 0:N-1;
x = zeros(N,1);
x(1) = 1;

% delay line
tau = 0.00025; % delay [seconds]
M = floor(tau*fs);
delayline = zeros(M, 1);

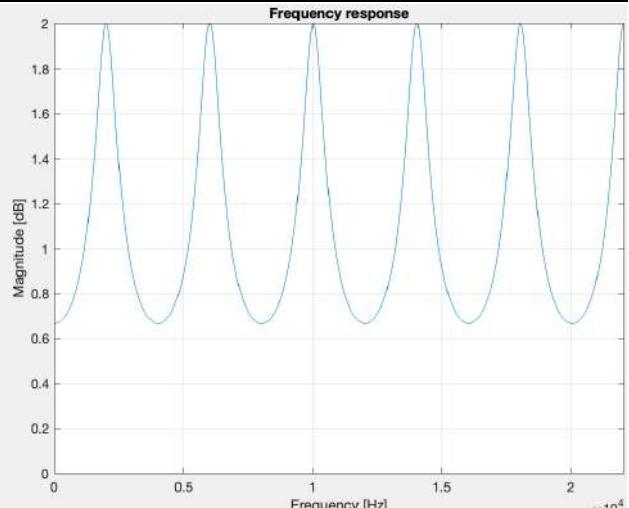
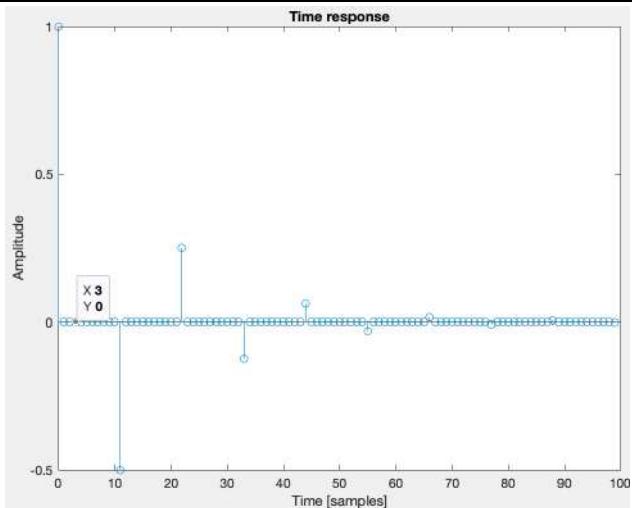
% delay gain
g = -0.5;
c = 1;

y = zeros(N, 1);
for ii = 1:N
    y(ii) = c*x(ii) + g*delayline(end);
    delayline = [y(ii); delayline(1:end-1)];
end

figure,
stem(n,y), xlabel('Time [samples]'), ylabel('Amplitude')
title('Time response')

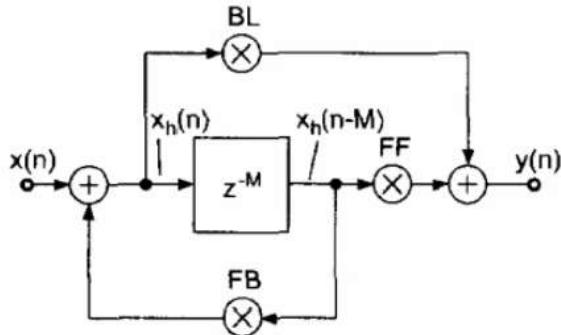
[H, f] = freqz(y, 1, 'half', fs);

figure,
plot(f*fs/pi/2, abs(H)), xlabel('Frequency [Hz]'), ylabel('Magnitude [dB]')
axis([0, fs/2, 0 2]), grid
title('Frequency response')
```



## Universal comb filter

We can merge together the two structures to obtain what is called universal comb filter.



By enabling or disabling the BL, FB and FF switches we

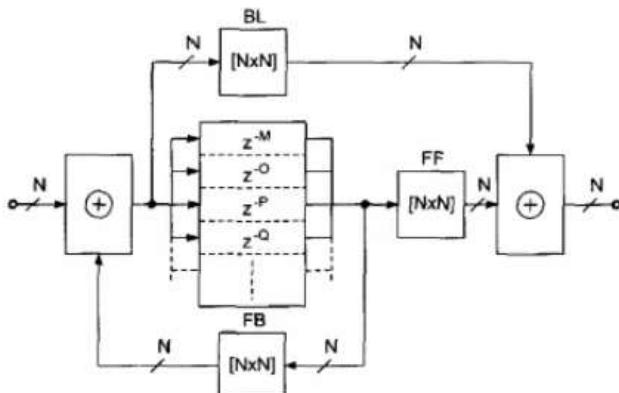
	BL	FB	FF
FIR comb filter	X	0	X
IIR comb filter	1	X	0
Allpass	a	-a	1
Delay	0	0	1

can obtain different type of filters.

We remind that the symbol X represents a don't care.

## Parallel connection of comb filters

We are ready to integrate together the small building blocks we have just seen. A real digital effect is implemented with a parallel connection of universal comb filters as we can see in the image below.



Since we have extended the simple comb filter to parallel connection of  $N$  comb filters, also the feedback, feedforward and blend coefficient become extended to  $N \times N$  matrices.

As we will see in the lecture on reverberation, most of the most famous effects (slapback, echo, reverb) are based on this structure.

## Delay-based audio effects

We talked about fractional delays, let see how to use them in two of the most famous effects:

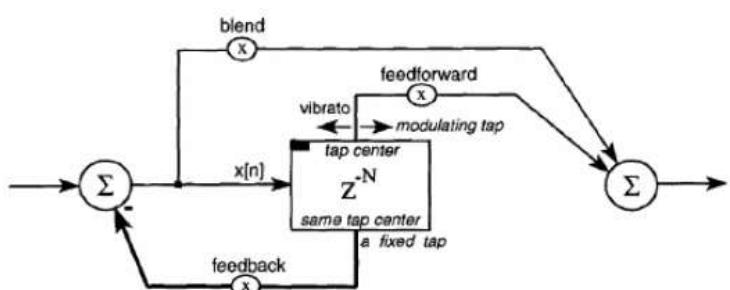
- *Chorus and flange effects*

Output signal is the sum of input signal and a delayed replica of the input signal. Delay is varied over time and the delay lines that generate it are fractional ones in order to avoid discontinuities in the delays. Each delay line is distinguished from the other by the minimum delay amount.

- *Vibrato*

The output signal is a delayed replica of the inputs signal. Delayed is varied over time sinusoidal fashion and there is no direct path that mesh the processed signal and the original one.

Many other effects are obtained with the same principles. The implementation of this multi-effect system proposed by J. Datorro is the one reported on the right. This is the industry standard structure and, as we can see, it is based on variable-length delay line.



To generate a certain effect, we have to configure differently the system according to the table below.

	BL	FF	FB	DELAY	DEPTH	MOD
Vibrato	0	1	0	0 ms	0 – 3 ms	0.1 – 5 Hz Sine
Flanger	0.7	0.7	0.7	0 ms	0 – 2 ms	0.1 – 1 Hz Sine
(white) Chorus	0.7	1	(–0.7)	1 – 30 ms	1 – 30 ms	Lowpass noise
Doubling	0.7	0.7	0	10 – 100 ms	1 – 100 ms	Lowpass noise

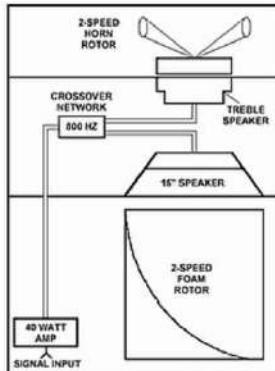
White chorus since it is modulated with a low-passed noise signal.

## The Hammond Organ

Let us see now a specific case: the rotating speaker of a Hammond organ.

The Hammond organ is an electromechanics device that wants to reproduce the pipe organ. Since it does not resample at all the sound of a pipe organ, they try to correct the timbre by adopting a particular kind of speaker called Leslie speaker.

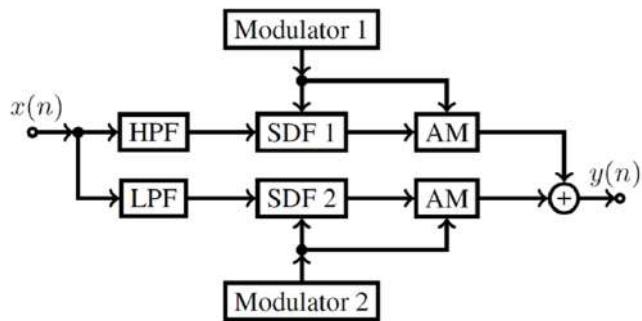
Let us see how it is built.



The treble loudspeaker is a rotating horn loudspeaker. Just one sounds, the other is there for symmetry and infertility reasons. Since horns are highly directive radiators, a fixed listener perceives a strong amplitude modulation. Furthermore, since the sound source is at the mouth of the horn, not in the centre of rotation, a fixed listener perceives also frequency modulation. The frequency modulation is nothing but the doppler effect for which we have

$$\omega_l = \omega_s \frac{1 + \frac{v_s l}{c}}{1 - \frac{v_s l}{c}}$$

The same is done for the bass but, in this case, the speaker is fixed, and a rotating system spreads the sound. This generated amplitude modulation due to rotation and directivity enhancement introduced by the drum.



If we would like to design a digital implementation of this speaker-filter, we come up with something like that. Let us dig into the details of each blocks: *Low pass and high pass filters*

The low pass and high pass filters are realisable with any suitable filter implementation. A possible choice could be to use a 4<sup>th</sup> order Butterworth IIR filter with cut-off frequency  $f_c = 800 \text{ Hz}$ . In digital implementation, is realised with a digital approximation of the actual crossover filter.

### Modulators

The modulators are two sinusoidal signals that match the movement of the rotors. We implement them by means of two separate oscillators that mimic the presence of two distinct motors. In order to resemble the effects of the two rotors, we have to use different modulator frequency envelopes:

- A first one that mimic the characteristic speed-up and slowdown.
- A second one that takes into account the fact that the bass unit accelerates slower due to the greater inertia of the unit.

As we know, the two rotors can spin at two different speeds that lead to two different modulation frequencies:

- Slow speed  $f_m = 2 \text{ Hz}$
- High speed  $f_m = 6 \text{ Hz}$

Modulator of high path is driven at a frequency  $0.1 \text{ Hz}$  higher than the bass path.

Simply speaking, there some switchers to change the rotation of the elements. We report some values as examples

	Treble unit rotation	Bass unit rotation
Chorale	50 rpm	40 rpm
Tremolo	400 rpm	340 rpm

### Amplitude Modulation

The amplitude modulation is simpler to implement if the frequency-dependent radiation pattern of the bass speaker and the horns are discarded. It is performed by simply multiplying the signal by an appropriately scaled sinusoid. The difference equation of the output-signal of the filter will be:

$$y(n) = [1 + \alpha m(n)]x(n)$$

Where, to get good results for both treble and bass paths we set  $\alpha = 0.9$

### Frequency modulation

The frequency modulation applies a frequency-dependent delay. It is efficiently implemented with a series of 1<sup>st</sup> order all-pass filters whose transfer function is

$$H(z) = \frac{a_1 + z^{-1}}{1 + a_1 z^{-1}} \quad \text{with } a_1 \in [-1, 1]$$

That has a magnitude response equal to 0 dB for all frequencies and a phase response that is, in general, non-linear. This phase is characterised by a group delay that can be written as

$$\tau_g(\omega) = -\frac{\partial \angle H(\omega)}{\partial \omega} = \frac{1 - a_1^2}{1 + 2a_1 \cos(\omega) + a_1^2}$$

Thus, the cascade of  $N$  1<sup>st</sup> order all-pass filter will be represented by

$$H_N(z) = \left( \frac{a_1 + z^{-1}}{1 + a_1 z^{-1}} \right)^N$$

Where  $|a_1| < 1$  to ensure stability.

Since the modulator changes the sound in time, the coefficient  $a_1(n)$  must be function of time

$$a_1(n) = m'(n)$$

The difference equation will be

$$y(n) = m'(n)x(n) + x(n-1) - m'(n)y(n-1)$$

Where  $m'(n)$ , for a modified modulator, is equal to

$$m'(n) = M_s m(n) + M_b$$

Parameter	Treble (SDF1)	Bass (SDF2)
SDF length $N$	4	3
Modulator scaler $M_s$	0.2	0.04
Modulator bias $M_b$	-0.75	-0.92

Obviously, the model can be made as complex as we want: we could, for instance, model also the cabinet and loudspeaker elements (possibly non-linear).

## Recap on maths

In today's lecture we will see the basic mathematical tools for sound analysis that we will use in the next lessons.

### Fourier Transform

The first definition we will see is the Fourier Transform for nondimensional signal of continuous variable  $x(t)$ . Let us see its definition:

$x: \mathbb{R} \rightarrow \mathbb{C}$

- Fourier Transform

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt, \quad \omega \in \mathbb{R}$$

- Inverse Fourier Transform

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega)e^{j\omega t} d\omega, \quad t \in \mathbb{R}$$

What is the practical information that the Fourier Transform returns of a certain signal  $x(t)$ ? How much each frequency of the signal resemble that specific sinusoid identified by  $\omega$ .

We remind that the space of the function is an Hilbert space, this means that the transform associated a function to a number.

### Multidimensional Fourier Transform

When we deal with multidimensional signal of continuous variables  $x(\underline{r})$ , we need a multidimensional Fourier Transform defined as

$x: \mathbb{R}^D \rightarrow \mathbb{C}$

- Fourier Transform

$$X(\underline{k}) = \int_{-\infty}^{\infty} x(\underline{r})e^{-j\langle \underline{k}, \underline{r} \rangle} d\underline{r}, \quad \underline{k} \in \mathbb{R}^D$$

- Inverse Fourier Transform

$$x(\underline{r}) = \left( \frac{1}{2\pi} \right)^D \int_{-\infty}^{\infty} X(\underline{k})e^{j\langle \underline{k}, \underline{r} \rangle} d\underline{k}, \quad \underline{r} \in \mathbb{R}^D$$

The vector  $\underline{r}$  is easier to be interpreted in space instead of in time. It can be, for instance, the vector representing the coordinates of a point in a 3D environment.

We remind that  $\langle \underline{k}, \underline{r} \rangle$  is the equivalent of the dot product in monodimensional domain.

## Fourier Series

When we deal with periodic signal (of period  $T$ ) we use the Fourier Series  
 $x: \mathbb{R} \rightarrow \mathbb{C}, x(t) = x(t - T)$

- Fourier Transform

$$X[k] = \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) e^{-j\left(\frac{2\pi}{T}\right)kt} dt, \quad k \in \mathbb{Z}$$

- Inverse Fourier Transform

$$x(t) = \sum_{k=-\infty}^{\infty} X[k] e^{j\left(\frac{2\pi}{T}\right)kt}, \quad t \in \left[-\frac{T}{2}, \frac{T}{2}\right]$$

As we can notice, the frequency space is discrete.

## Acoustic Field

The acoustic field is a real-valued scalar function, hereafter called  $p(\underline{r}, t)$ , whose domain extends in

- Time  $t \in \mathbb{R}$
- Space  $\underline{r} \in \mathbb{R}^3$

Therefore, the acoustic field is 4-dimensional space since we have one dimension for time and the three dimensions for space.

One important property is that this acoustic field is invariant under change of spatial coordinates: it does not depend on the coordinates we use to represent it. This does not imply that the mathematical expression remains the same: we can represent a same acoustic field with different equations. This happens, for example, when we change the coordinate system.

## Differential operator

Let us now go over a brief recap on the differential operator.

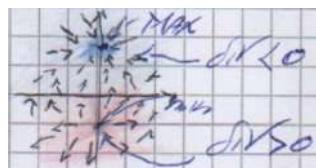
- *Gradient Operator*

The gradient operator  $\nabla$  is a vector differential operator. It means that it is applied over a scalar and it returns a vector. Considering a three-dimensional space expressed with Cartesian coordinates, i.e.,  $\underline{r} = [x, y, z]^T$ , we define the gradient of the acoustic field as

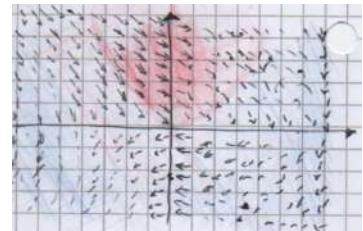
$$\nabla p(\underline{r}, t) = \left[ \frac{\partial p(\underline{r}, t)}{\partial x}, \frac{\partial p(\underline{r}, t)}{\partial y}, \frac{\partial p(\underline{r}, t)}{\partial z} \right]^T$$

- *Divergence Operator*

The divergence operator is instead a differential operator. It works over a vector field and produces a scalar. Considering a vector field  $\underline{F} = [F_x, F_y, F_z]^T$ , we have that



$$\text{div}(\underline{F}) = \nabla \cdot \underline{F} = \frac{\partial F_x}{\partial x} + \frac{\partial F_y}{\partial y} + \frac{\partial F_z}{\partial z}$$



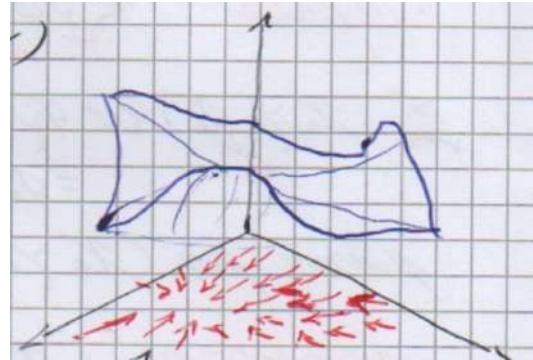
- **Laplace Operator**

The Laplace operator  $\nabla^2$  describes the curvature of the function in space. It operates on a scalar field, producing a scalar field. It is defined as the divergence of the gradient operator.

$$\nabla^2 p(\underline{r}, t) = \nabla \cdot (\nabla p(\underline{r}, t))$$

Takes different forms according to the coordinate system adopted to represent  $\underline{r}$ . Its representation in Cartesian coordinates is:

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$$



### **Homogenous wave equation and homogeneous Helmholtz equation**

We consider a space where there are no sources and where we consider ONLY the linear oscillations. In that condition the acoustic field can be described by the homogenous wave equation

$$\nabla^2 p(\underline{r}, t) - \frac{1}{c^2} \frac{\partial^2 p(\underline{r}, t)}{\partial t^2} = 0$$

Let us cast the problem in frequency domain. By assuming the time-harmonic behaviour (so the fact that the equation can be written as  $p(\underline{r}, t) = P(\underline{r}, \omega) e^{-j\omega t}$ ) we can find the Fourier transform of the harmonic wave equation. It is called the **homogenous Helmholtz Equation**

$$\nabla^2 P(\underline{r}, \omega) + \left(\frac{\omega}{c}\right)^2 P(\underline{r}, \omega) = 0$$

$P(\underline{r}, t)$  refers to the amplitude of the pressure at a specific time  $\underline{r}$  and at specific frequency  $\omega$ .

Let us try to find the solution of this equation in Cartesian coordinates. It turns out that it is nothing but

$$P(\underline{r}, \omega) = e^{j\langle \underline{k}, \underline{r} \rangle}$$

Where the vector  $\underline{k}$  that appears in the inner product is called wavenumber vector  $\underline{k} = [k_x, k_y, k_z]^T$

Furthermore, complex exponential function is a solution only if the wavenumber vector satisfies the **dispersion relation**:

$$\|\underline{k}\|^2 = \left(\frac{\omega}{c}\right)^2 \quad \left[\frac{\text{rad}}{\text{m}}\right]$$

It is easy to verify that  $P(\underline{r}, \omega) = e^{j\langle \underline{k}, \underline{r} \rangle}$  is a solution of the Helmholtz equation. If we plug it there, it becomes equal to the norm of the wavenumber vector. That solution is called plane wave. There are two kinds of plane waves: propagating plane waves and evanescent plane waves.

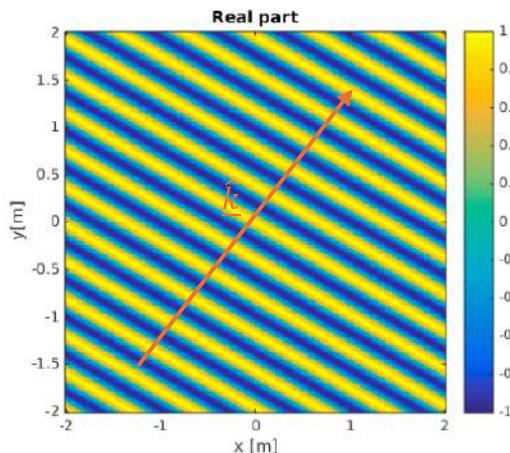
- *Propagating plane waves*

In a four-dimensional domain  $\underline{k} \in \mathbb{R}^3, t \in \mathbb{R}$  points at constant phase have constant inner product

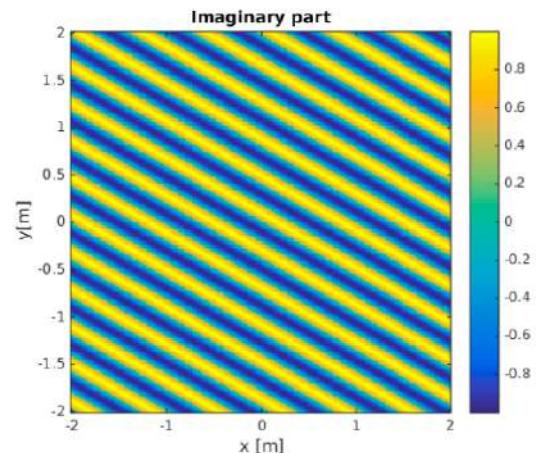
$$\angle P(\underline{r}, \omega) = kost \quad \Rightarrow \quad \langle \underline{k}, \underline{r} \rangle = C, C \in \mathbb{R}$$

We can think at them as planes orthogonal to  $\underline{k}$ . We identify the **direction of arrival** with the unit vector  $\hat{\underline{k}}$  relative to the vector  $\underline{k}$ . It is defined as  $\hat{\underline{k}} = \frac{\underline{k}}{\|\underline{k}\|}$

In a figure we get (assuming the time frozen)

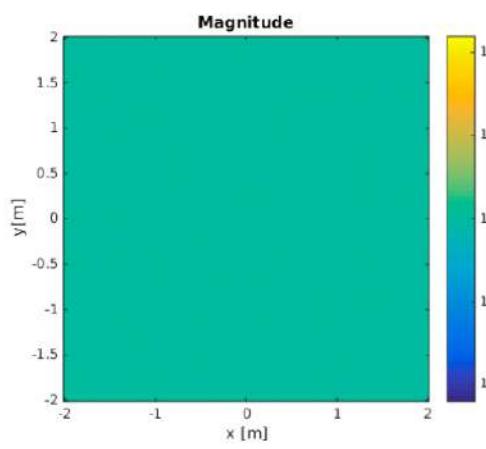


(a) Real part.

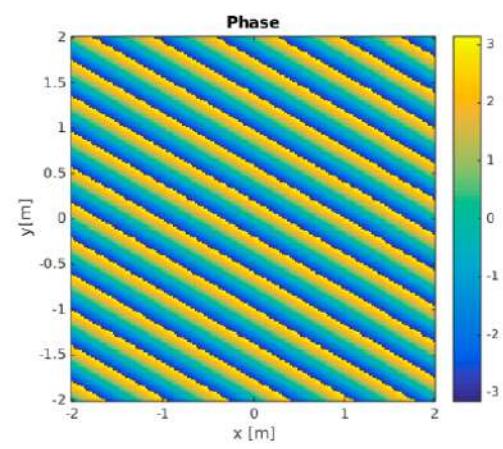


(b) Imaginary part.

By representing the equation in polar coordinates, we can see that the magnitude is constant in every point of the space.



(a) Magnitude.



(b) Phase.

- *Evanescence plane waves*

Let assume that  $k_x, k_y \in \mathbb{R}$ . Let assume also that we can write

$$\|\underline{k}\|^2 = k_x^2 + k_y^2 + k_z^2 = \left(\frac{\omega}{c}\right)^2$$

Assuming that we know  $k_x^2 + k_y^2$  we rewrite  $k_z$  in function of them

$$k_z = \sqrt{\left(\frac{\omega}{c}\right)^2 - k_x^2 - k_y^2}$$

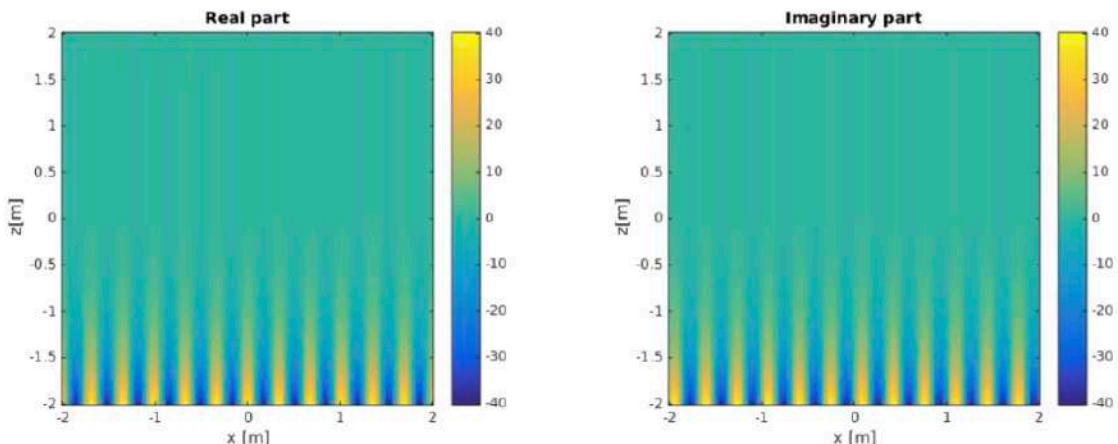
Two cases:

- 1)  $k_x^2 + k_y^2 \leq \left(\frac{\omega}{c}\right)^2 \rightarrow k_z \in \mathbb{R}$ : propagating plane waves (case seen before)
- 2)  $k_x^2 + k_y^2 > \left(\frac{\omega}{c}\right)^2 \rightarrow k_z \in \mathbb{C}$ : evanescent plane waves

In this case  $k_z = j\zeta, \zeta \in \mathbb{R}^+$

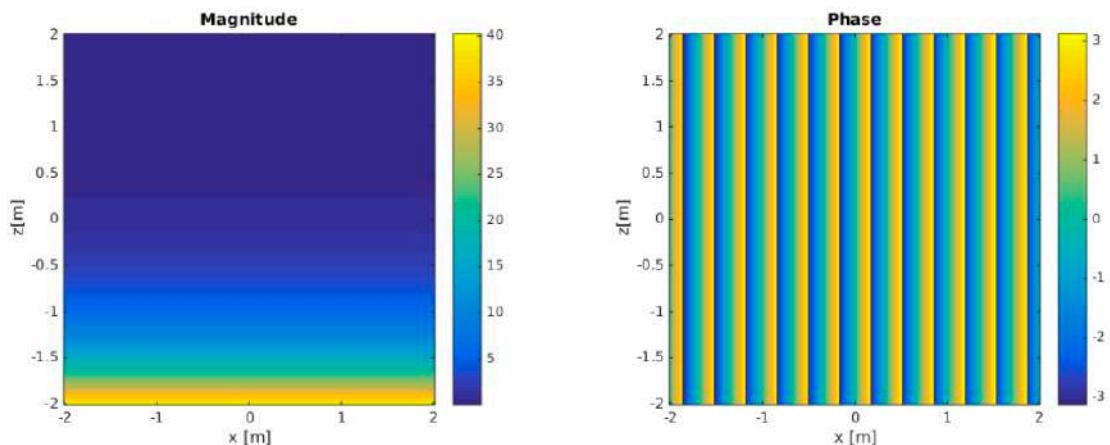
This means that we can write the evanescent waves as

$$P(\underline{r}, \omega) = e^{j(k_x x + k_y y + j\zeta z)} = e^{j(k_x x + k_y y)} e^{-\zeta z}$$



(a) Real part.

(b) Imaginary part.



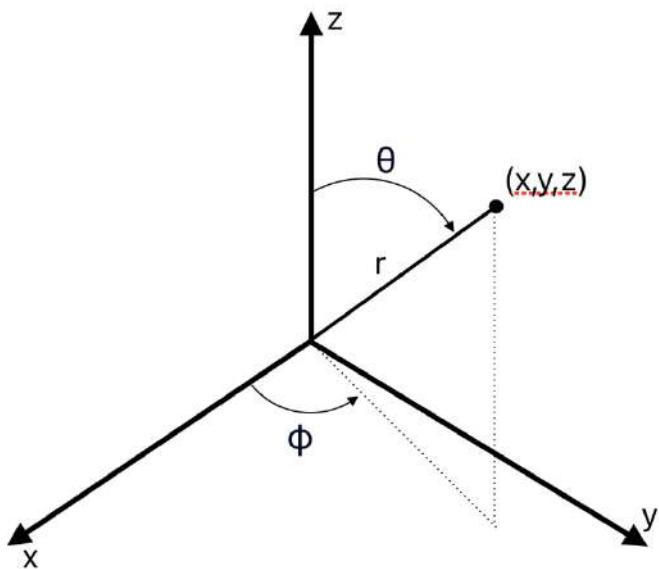
(a) Magnitude.

(b) Phase.

As we can see, the magnitude is exponentially decaying. The phase stays unchanged!

## From Cartesian to spherical coordinates

Let us see how things can become a little more messy.



Cartesian coordinates  $(x, y, z)$  are expressed as functions of spherical coordinates (radius  $r$ , co-elevation  $\theta$ , azimuth  $\phi$ ) by

$$\begin{cases} x = r \sin(\theta) \cos(\phi) \\ y = r \sin(\theta) \sin(\phi) \\ z = r \cos(\theta) \end{cases}$$

The inverse mapping is

$$\begin{cases} r = \sqrt{x^2 + y^2 + z^2} \\ \theta = \arccos\left(\frac{z}{r}\right) \\ \phi = \arctan\left(\frac{y}{x}\right) \end{cases}$$

If we make this conversion, we have that the Laplace operator can be expressed as:

$$\nabla^2 = \frac{1}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial}{\partial r} \right) + \frac{1}{r^2 \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial}{\partial \theta} \right) + \frac{1}{r^2 \sin^2(\theta)} \frac{\partial^2}{\partial \phi^2}$$

Therefore, the Helmholtz equation becomes:

$$\frac{1}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial P}{\partial r} \right) + \frac{1}{r^2 \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial P}{\partial \theta} \right) + \frac{1}{r^2 \sin^2(\theta)} \frac{\partial^2 P}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 P = 0$$

Let us try to reason on how we can find a solution to this equation. We can write the solution as a product of functions that depends on just one single coordinate.

$$P(r, \omega) = R(r)\Theta(\theta)\Phi(\phi)$$

Where:

- $R(r)$  is the radial dependency
- $\Theta(\theta)$  and  $\Phi(\phi)$  are the angular dependencies

### Reminder: variation of constants method

In mathematics, variation of parameters, also known as variation of constants, is a general method to solve inhomogeneous linear ordinary differential equations.

For first-order inhomogeneous linear differential equations it is usually possible to find solutions via integrating factors or undetermined coefficients with considerably less effort, although those methods leverage heuristics that involve guessing and do not work for all inhomogeneous linear differential equations. Simply speaking, we can express the solution of differential equation as the product of function that are, in turn, function of some parameters. This makes it easier to find the solution since we, quite often, the parameters are chosen in a smart way. Furthermore, we can solve the equation step by step analysing one parameter at a time.

If we substitute the solution in the Helmholtz equation, we get

$$\frac{1}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial P}{\partial r} \right) + \frac{1}{r^2 \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial P}{\partial \theta} \right) + \frac{1}{r^2 \sin^2(\theta)} \frac{\partial^2 P}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 P = 0$$

$$\begin{aligned} & \frac{1}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)\Theta(\theta)\Phi(\phi)}{\partial r} \right) + \frac{1}{r^2 \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial R(r)\Theta(\theta)\Phi(\phi)}{\partial \theta} \right) + \\ & + \frac{1}{r^2 \sin^2(\theta)} \frac{\partial^2 R(r)\Theta(\theta)\Phi(\phi)}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 R(r)\Theta(\theta)\Phi(\phi) = 0 \end{aligned}$$

Let us now focus on the dependency on azimuth:

$$\begin{aligned} & \Theta(\theta)\Phi(\phi) \frac{1}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{R(r)\Phi(\phi)}{r^2 \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) + \\ & + \frac{R(r)\Theta(\theta)}{r^2 \sin^2(\theta)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 R(r)\Theta(\theta)\Phi(\phi) = 0 \end{aligned}$$

$$\begin{aligned} & \frac{r^2}{R(r)\Theta(\theta)\Phi(\phi)} \left[ \Theta(\theta)\Phi(\phi) \frac{1}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{R(r)\Phi(\phi)}{r^2 \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) \right. \\ & \left. + \frac{R(r)\Theta(\theta)}{r^2 \sin^2(\theta)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 R(r)\Theta(\theta)\Phi(\phi) \right] = 0 \cdot \frac{r^2}{R(r)\Theta(\theta)\Phi(\phi)} \end{aligned}$$

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{1}{\Theta(\theta) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) + \frac{1}{\Phi(\phi) \sin^2(\theta)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

Since we want to focus on the dependency on azimuth, we treat all the other terms as a global constant that we call  $m^2$  (the square will be useful to simplify the expression of the solution of the differential equation).

$$\frac{1}{\Phi(\phi)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} = -\sin^2(\theta) \left[ \frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{1}{\Theta(\theta) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) + \left( \frac{\omega}{c} \right)^2 r^2 \right] = -m^2$$

Therefore, the equation is reduced to:

$$\frac{1}{\Phi(\phi)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} = -m^2 \rightarrow \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} = -m^2 \Phi(\phi)$$

The solution of an equation of this shape is:

$$\Phi(\phi) = e^{jm\phi}$$

Reminder: solution of a second order homogeneous linear equation with constant coefficients  
 First, it is homogeneous since the known term is equal to zero, linear since the function appears only at its first grade, constant coefficients since the coefficient before the function does not depends on  $\phi$ .

An equation like this one can be written as

$$y''(t) + a_1(t) + a_0 y(t) = 0 \quad \text{with } a_1, a_0 \in \mathbb{R}$$

In our specific case,  $a_1 = 0, a_0 = m^2$

To solve this equation, we look for its characteristic polynomial  $P(\lambda)$  and we find its solution  $\lambda_1$  and  $\lambda_2$ .

$$\lambda^2 + a_1\lambda + a_0 = 0$$

Depending on the  $\Delta$ , we get different solutions:

$\Delta > 0$ $\lambda_1, \lambda_2 \in \mathbb{R}$	$y(t) = c_1 e^{\lambda_1 t} + c_2 e^{\lambda_2 t}$ $c_1, c_2 \in \mathbb{R}$
$\Delta = 0$ $\lambda_1 = \lambda_2 \in \mathbb{R}$	$y(t) = c_1 e^{\lambda_1 t} + c_2 t e^{\lambda_2 t}$ $c_1, c_2 \in \mathbb{R}$
$\Delta < 0$ $\lambda_1 = \alpha + i\beta, \lambda_2 = \alpha - i\beta \in \mathbb{C}$	$y(t) = c_1 e^{\alpha t} \cos(\beta t) + c_2 e^{\alpha t} \sin(\beta t)$ $c_1, c_2 \in \mathbb{R}$

In our specific case we will have

$$\frac{\partial^2 \Phi(\phi)}{\partial \phi^2} + m^2 \Phi(\phi) = 0$$

$$\lambda^2 + m^2 = 0 \rightarrow \lambda = -m^2 \xrightarrow{\Delta < 0} \lambda_{1,2} = 0 \pm jm$$

$$\Phi(\phi) = c_1 e^{0t} \cos(m\phi) + c_2 e^{0t} \sin(m\phi) = c_1 \cos(m\phi) + c_2 \sin(m\phi) = ? e^{jm\phi}$$

We assume, just for convenience,  $m \in \mathbb{Z}$  and we substitute the solution in the general equation.

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{1}{\Theta(\theta) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) + \frac{1}{\Phi(\phi) \sin^2(\theta)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{1}{\Theta(\theta) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) + \frac{1}{e^{jm\phi} \sin^2(\theta)} \frac{\partial^2 e^{jm\phi}}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{1}{\Theta(\theta) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) + \frac{-m^2 e^{jm\phi}}{e^{jm\phi} \sin^2(\theta)} + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{1}{\Theta(\theta) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) - \frac{m^2}{\sin^2(\theta)} + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

Same procedure we have just done, but for  $\Theta(\theta)$ . Once again, the coefficient is set equal to  $l(l + 1)$  just for convenience.

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{1}{\Theta(\theta) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) - \frac{m^2}{\sin^2(\theta)} + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

$$\frac{1}{\Theta(\theta) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) = - \underbrace{\left[ \frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \left( \frac{\omega}{c} \right)^2 r^2 \right]}_{l(l+1)} + \frac{m^2}{\sin^2(\theta)} = -l(l+1) + \frac{m^2}{\sin^2(\theta)}$$

$$\frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial}{\partial \theta} \right) \Theta(\theta) = - \left[ l(l+1) - \frac{m^2}{\sin^2(\theta)} \right] \sin(\theta) \Theta(\theta)$$

$$\frac{\partial}{\partial \theta} \left( (1 - \cos^2(\theta)) \frac{\partial}{\partial \theta} \right) \Theta(\theta) = - \left[ l(l+1) \sin(\theta) - \frac{m^2}{\sin(\theta)} \right] \Theta(\theta)$$

The equation written above can be recognised as the associated Legendre equation, whose solution is:

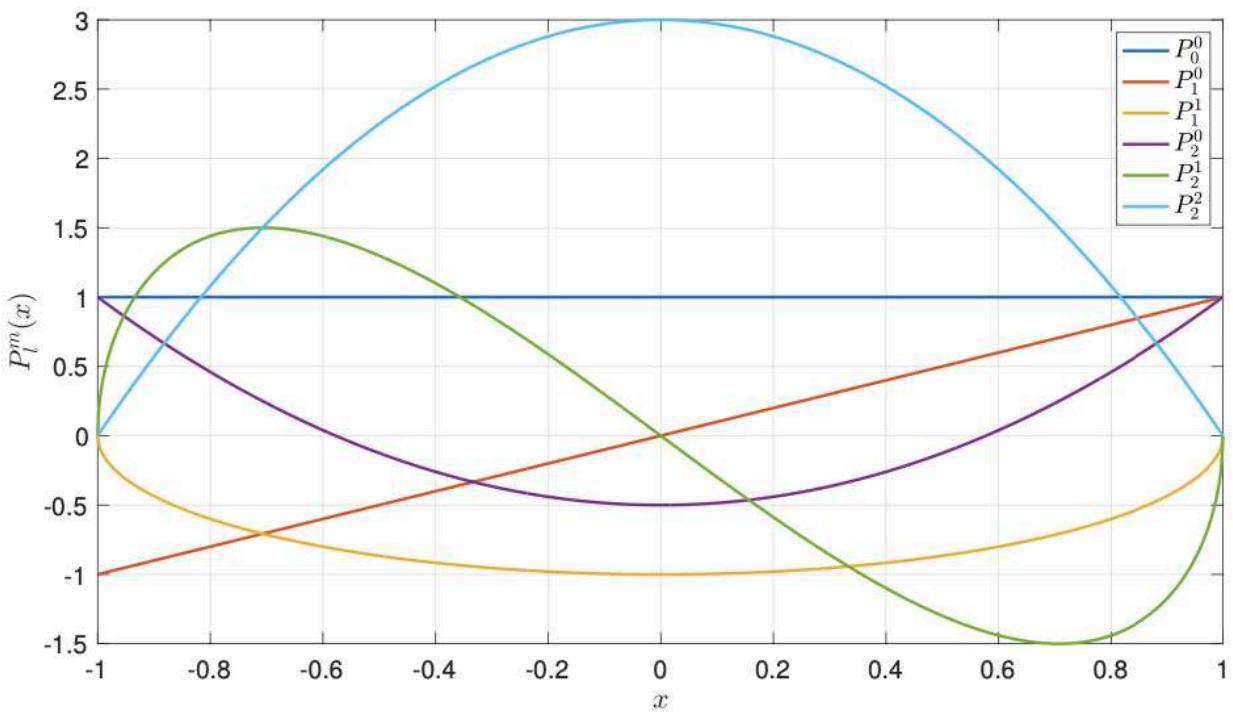
$$\Theta(\theta) = P_l^m(\cos(\theta))$$

Where:

- $P_l^m(\cdot)$  is the associated Legendre polynomial of integer order  $l$  and degree  $m$
- $l = 0, 1, \dots$  and  $m = -l, \dots, l$

Out of curiosity let us see which the solution of the Legendre polynomials associated to order and grades are.

Adrien-Marie Legendre



Since  $\Theta(\theta) = P_l^m(\cos(\theta))$  is function of a cosine, they are represented on a domain that goes from  $-1$  to  $1$ .

Reminder (ma mica tanto): Legendre's differential equation

The Legendre's differential equation can be written as

$$\frac{d}{dx} \left( (1 - x^2) \frac{d}{dx} \right) P(x) = -\lambda P(x)$$

with the eigenvalue  $\lambda$  in lieu of  $l(l + 1)$ .

We have found  $\Phi(\phi)$  and  $\Theta(\theta)$ . Let us merge together into a single function called **spherical harmonic** function:

$$Y_l^m(\theta, \phi) = (-1)^m \sqrt{\frac{(2l+1)}{4\pi} \frac{(l-|m|)!}{(l+|m|)!}} e^{jm\phi} P_l^{|m|}(\cos(\theta))$$

This function has lot of interesting properties:

- Conjugation

The conjugate of the spherical harmonic is just the spherical harmonic itself with opposite degree.

$$(Y_l^m(\theta, \phi))^* = Y_{-l}^{-m}(\theta, \phi)$$

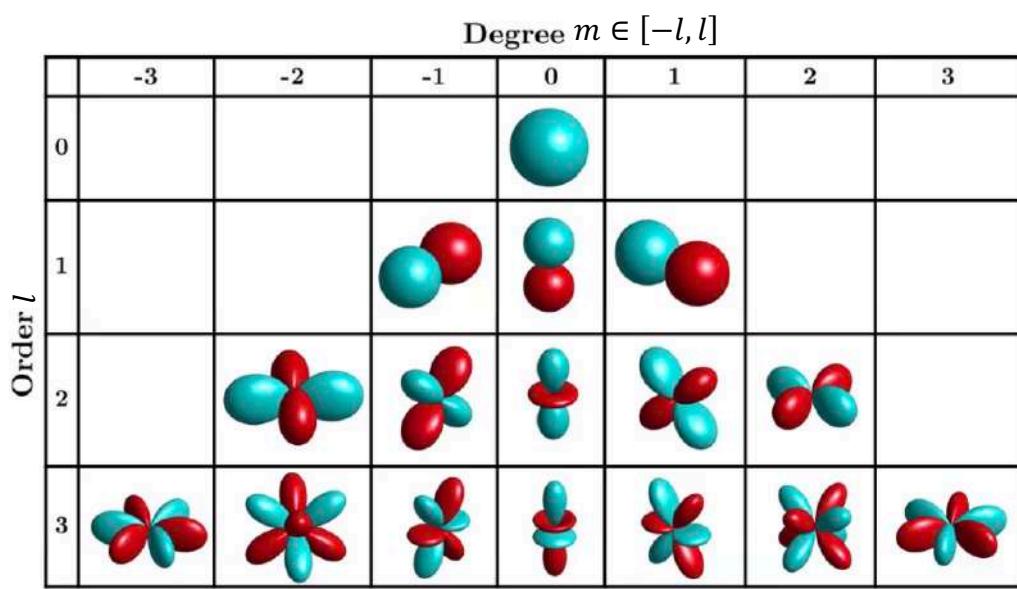
- Orthogonality and normalisation

It means that we have a value equal to one where order and degree are the same. This is represented by two Kronecker deltas.

$$\int_0^{2\pi} \int_0^\pi Y_l^{-m}(\theta, \phi) Y_{l'}^{m'}(\theta, \phi) \sin(\theta) d\theta d\phi = \delta[l - l'] \delta[m - m']$$

Thanks to these properties they are indeed basic function of a spherical Hilbert Space: every function can be represented as a linear combination of them. Last remark, they are frequency independent!

Let us see these bases.



First row, monopole.  
Second row, dipoles.  
As we can see it is obvious that they are orthogonal one between the others.  
If we increase the order, the interpretation becomes a little bit floured.

Let us now focus on the dependency on radius starting from the general expression and substituting there the equation of  $\Phi(\phi) = e^{jm\phi}$  and  $\Theta(\theta) = P_l^m(\cos(\theta))$ :

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{1}{\Theta(\theta) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) + \frac{1}{\Phi(\phi) \sin^2(\theta)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{1}{\Theta(\theta) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial \Theta(\theta)}{\partial \theta} \right) + \frac{1}{e^{jm\phi} \sin^2(\theta)} \frac{\partial^2 e^{jm\phi}}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \frac{1}{P_l^m(\cos(\theta)) \sin(\theta)} \frac{\partial}{\partial \theta} \left( \sin(\theta) \frac{\partial P_l^m(\cos(\theta))}{\partial \theta} \right) - \frac{m^2}{\sin^2(\theta)} + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) - l(l+1) + \frac{m^2}{\sin^2(\theta)} - \frac{m^2}{\sin^2(\theta)} + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

$$\frac{1}{R(r)} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) - l(l+1) + \left( \frac{\omega}{c} \right)^2 r^2 = 0$$

$$\frac{\partial}{\partial r} \left( r^2 \frac{\partial R(r)}{\partial r} \right) + \left[ \left( \frac{\omega}{c} \right)^2 r^2 - l(l+1) \right] R(r) = 0$$

This equation has two different solution:

- In terms of **Spherical Bessel functions**

$$R(r) = R_1 j_l \left( \left( \frac{\omega}{c} \right) r \right) + R_2 y_l \left( \left( \frac{\omega}{c} \right) r \right)$$

- In terms of **Spherical Hankel functions**

$$R(r) = R_3 h_l^{(1)} \left( \left( \frac{\omega}{c} \right) r \right) + R_4 h_l^{(2)} \left( \left( \frac{\omega}{c} \right) r \right)$$

Where:

$$h_l^{(1)} \left( \left( \frac{\omega}{c} \right) r \right) = j_l \left( \left( \frac{\omega}{c} \right) r \right) + jy_l \left( \left( \frac{\omega}{c} \right) r \right)$$

$$h_l^{(2)} \left( \left( \frac{\omega}{c} \right) r \right) = j_l \left( \left( \frac{\omega}{c} \right) r \right) - jy_l \left( \left( \frac{\omega}{c} \right) r \right)$$

As we can notice, The Hankel functions are nothing but a combination of Bessel functions.  $j_l$  is the  $l$ -th order spherical Bessel function of the first type while the other  $y_l$  are Bessel functions of the second type.

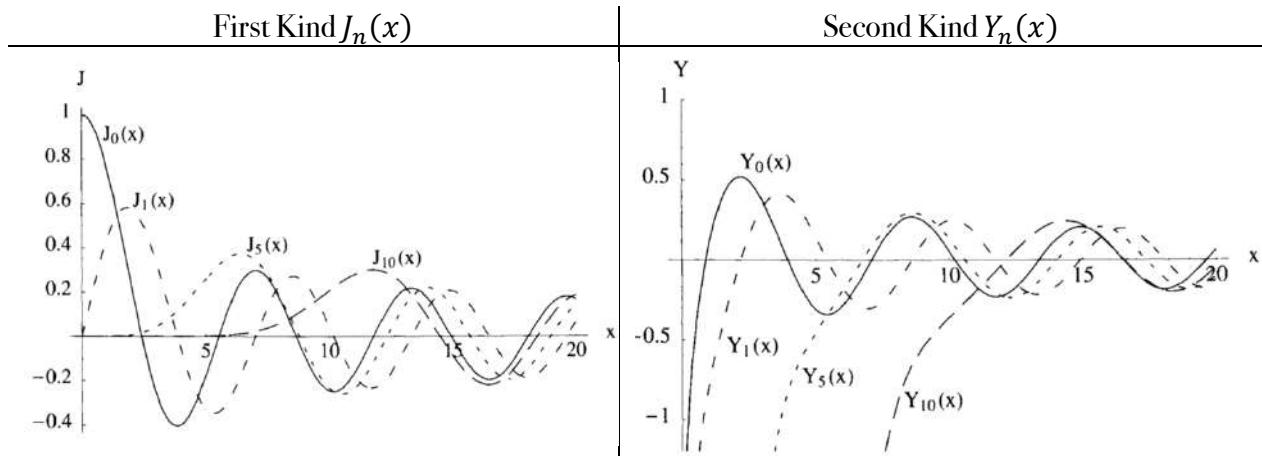
**Reminder: Bessel functions**

Bessel functions, first defined by the mathematician Daniel Bernoulli and then generalized by Friedrich Bessel, are canonical solutions  $y(x)$  of Bessel's differential equation:

$$x^2 \frac{d^2y}{dx^2} + x \frac{dy}{dx} + (x^2 - \alpha^2)y = 0$$

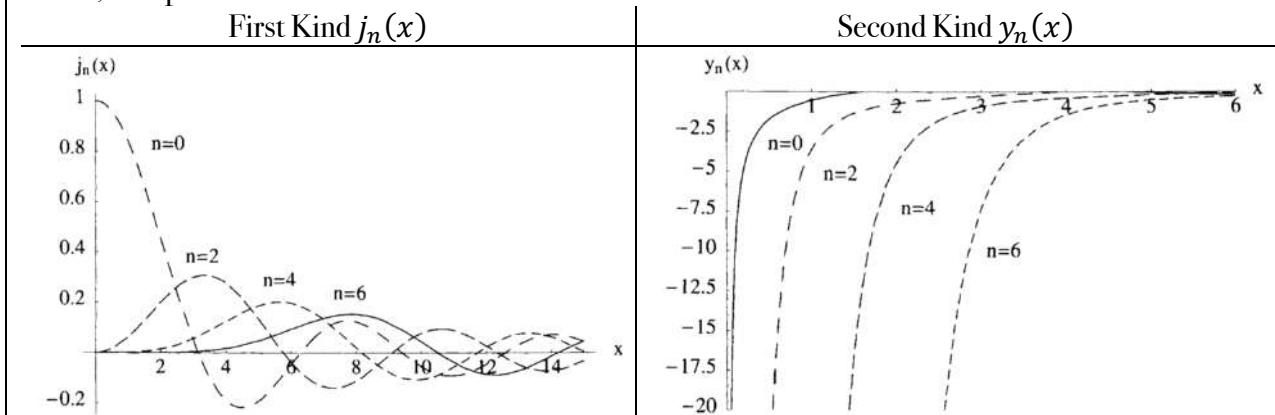
for an arbitrary complex number  $\alpha$ , which represents the order of the Bessel function.

These are the Bessel functions:



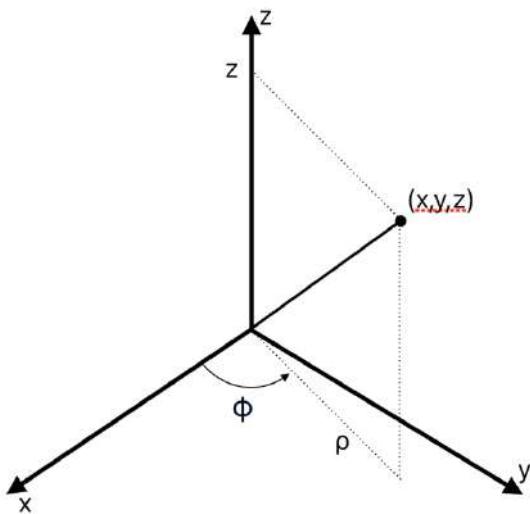
A subset of the Bessel functions are the so-called spherical Bessel functions that are nothing but the solution of the Bessel differential equation expressed in spherical coordinates.

Below, the spherical Bessel functions:



We notice that the value of the functions decreases with the order. So, the amplitude of the spherical oscillation will be higher for low value of  $n$ .

## Cylindrical coordinates



Let us see how the things changes in cylindrical coordinates where a point is identified by (radial distance  $\rho$ , azimuth  $\phi$ , height  $z$ ). Let us express them as function of spherical coordinates by

$$\begin{cases} \rho = r \sin(\theta) \\ \phi = \phi \\ z = r \cos(\theta) \end{cases}$$

In cylindrical coordinates, the parameter  $z$  is shared with the cartesian coordinates while  $\phi$  is shared with spherical ones.

We start again from the beginning looking at the expression of the Laplace operator:

$$\nabla^2 = \frac{\partial^2}{\partial \rho^2} + \frac{1}{\rho} \frac{\partial}{\partial \rho} + \frac{1}{\rho^2} \frac{\partial^2}{\partial \phi^2} + \frac{\partial^2}{\partial z^2}$$

Let us plug it into the expression of the Helmholtz equation:

$$\nabla^2 P(\underline{r}, \omega) + \left(\frac{\omega}{c}\right)^2 P(\underline{r}, \omega) = 0 \rightarrow \frac{\partial^2 P(\underline{r}, \omega)}{\partial \rho^2} + \frac{1}{\rho} \frac{\partial P(\underline{r}, \omega)}{\partial \rho} + \frac{1}{\rho^2} \frac{\partial^2 P(\underline{r}, \omega)}{\partial \phi^2} + \frac{\partial^2 P(\underline{r}, \omega)}{\partial z^2} + \left(\frac{\omega}{c}\right)^2 P(\underline{r}, \omega) = 0$$

Once again, solutions to the homogeneous Helmholtz equation in cylindrical coordinates can be obtained by separation of variables.

$$P(\underline{r}, \omega) = R(\rho)\Phi(\phi)Z(z)$$

Where:

- $R(\rho)$  is the radial dependency;
- $\Phi(\phi)$  is the azimuthal dependency.

By substituting the solution into the Helmholtz equation we get

$$\frac{\partial^2 P(\underline{r}, \omega)}{\partial \rho^2} + \frac{1}{\rho} \frac{\partial P(\underline{r}, \omega)}{\partial \rho} + \frac{1}{\rho^2} \frac{\partial^2 P(\underline{r}, \omega)}{\partial \phi^2} + \frac{\partial^2 P(\underline{r}, \omega)}{\partial z^2} + \left(\frac{\omega}{c}\right)^2 P(\underline{r}, \omega) = 0$$

$$\frac{\partial^2 R(\rho)\Phi(\phi)Z(z)}{\partial \rho^2} + \frac{1}{\rho} \frac{\partial R(\rho)\Phi(\phi)Z(z)}{\partial \rho} + \frac{1}{\rho^2} \frac{\partial^2 R(\rho)\Phi(\phi)Z(z)}{\partial \phi^2} + \frac{\partial^2 R(\rho)\Phi(\phi)Z(z)}{\partial z^2} + \left(\frac{\omega}{c}\right)^2 R(\rho)\Phi(\phi)Z(z) = 0$$

$$\frac{\Phi(\phi)Z(z)}{R(\rho)\Phi(\phi)Z(z)} \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{\Phi(\phi)Z(z)}{\rho R(\rho)\Phi(\phi)Z(z)} \frac{\partial R(\rho)}{\partial \rho} + \frac{R(\rho)Z(z)}{\rho^2 R(\rho)\Phi(\phi)Z(z)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} + \frac{R(\rho)\Phi(\phi)}{R(\rho)\Phi(\phi)Z(z)} \frac{\partial Z(z)}{\partial z^2} + \left(\frac{\omega}{c}\right)^2 \frac{R(\rho)\Phi(\phi)Z(z)}{R(\rho)\Phi(\phi)Z(z)} = 0 \cdot \frac{1}{R(\rho)\Phi(\phi)Z(z)}$$

$$\frac{1}{R(\rho)} \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{1}{\rho R(\rho)} \frac{\partial R(\rho)}{\partial \rho} + \frac{1}{\rho^2 \Phi(\phi)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} + \frac{1}{Z(z)} \frac{\partial Z(z)}{\partial z^2} + \left(\frac{\omega}{c}\right)^2 = 0$$

Let us see which is the dependency on  $z$  by considering constant the other terms:

$$\frac{1}{Z(z)} \frac{\partial Z(z)}{\partial z^2} = - \left[ \frac{1}{R(\rho)} \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{1}{\rho R(\rho)} \frac{\partial R(\rho)}{\partial \rho} + \frac{1}{\rho^2 \Phi(\phi)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} + \left( \frac{\omega}{c} \right)^2 \right] = -k_z^2$$

Once again, we have a second order homogeneous linear equation whose solution is

$$Z(z) = e^{jk_z z}$$

Therefore, the expression in function only of  $(\rho, \phi)$  will be

$$\frac{1}{R(\rho)} \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{1}{\rho R(\rho)} \frac{\partial R(\rho)}{\partial \rho} + \frac{1}{\rho^2 \Phi(\phi)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} + \underbrace{\frac{1}{Z(z)} \frac{\partial Z(z)}{\partial z^2}}_{-k_z^2} + \left( \frac{\omega}{c} \right)^2 = 0$$

$$\frac{1}{R(\rho)} \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{1}{\rho R(\rho)} \frac{\partial R(\rho)}{\partial \rho} + \frac{1}{\rho^2 \Phi(\phi)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} = -\left( \frac{\omega}{c} \right)^2 + k_z^2 = -k_\rho^2$$

From the last equality, we can write  $k_\rho^2 = \sqrt{\left( \frac{\omega}{c} \right)^2 - k_z^2}$

Let us now analyse, specifically, the dependence on  $\phi$ . We start from what we have just found

$$\frac{1}{R(\rho)} \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{1}{\rho R(\rho)} \frac{\partial R(\rho)}{\partial \rho} + \frac{1}{\rho^2 \Phi(\phi)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} = -k_\rho^2$$

We rearrange the equation, and we multiply everything by  $\rho^2$

$$\frac{\rho^2}{R(\rho)} \left( \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{1}{\rho} \frac{\partial R(\rho)}{\partial \rho} \right) + k_\rho^2 \rho^2 = -\frac{\rho^2}{\rho^2 \Phi(\phi)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2}$$

We consider all the terms that does not depend on  $\phi$  as constants (as they actually are if we considerer just the dependence on  $\phi$ )

$$\frac{1}{\Phi(\phi)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} = - \left[ \frac{\rho^2}{R(\rho)} \left( \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{1}{\rho} \frac{\partial R(\rho)}{\partial \rho} \right) + k_\rho^2 \rho^2 \right] = -m^2$$

Once again, our beloved second order homogeneous linear equation with constant coefficients whose solution is:

$$\Phi(\phi) = e^{jm\phi}$$

Finally, the dependence on  $\rho$ .

$$\begin{aligned} \frac{\rho^2}{R(\rho)} \left( \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{1}{\rho} \frac{\partial R(\rho)}{\partial \rho} \right) + k_\rho^2 \rho^2 &= - \frac{\rho^2}{\rho^2 \Phi(\phi)} \frac{\partial^2 \Phi(\phi)}{\partial \phi^2} = -m^2 \\ \left( \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{1}{\rho} \frac{\partial R(\rho)}{\partial \rho} \right) + k_\rho^2 R(\rho) &= - \frac{m^2}{\rho^2} R(\rho) \\ \frac{\partial^2 R(\rho)}{\partial \rho^2} + \frac{1}{\rho} \frac{\partial R(\rho)}{\partial \rho} + \left( k_\rho^2 + \frac{m^2}{\rho^2} \right) R(\rho) &= 0 \end{aligned}$$

Pop pop, it's show time, show time, show time, show time! Guess who's back again? Oh they don't know?  
Go on tell 'em! Oh they don't know? Go on tell 'em! It's the Bessel's equation walking in!

As we know, we can express its solutions in terms of

- Linear combination of Bessel functions (beware, this time we deal with the Bessel function not with the spherical ones).

$$R(\rho) = R_1 J_m(k_\rho \rho) + R_2 Y_m(k_\rho \rho)$$

- Linear combination of Hankel functions

$$R(\rho) = R_3 H_m^{(1)}(k_\rho \rho) + R_2 Y_m H_m^{(2)}(k_\rho \rho)$$

Where:

$$H_m^{(1)}(k_\rho \rho) = J_m(k_\rho \rho) + j Y_m(k_\rho \rho)$$

$$H_m^{(2)}(k_\rho \rho) = J_m(k_\rho \rho) - j Y_m(k_\rho \rho)$$

As we can see in the reminder, the behaviour of the first kind of the Bessel functions is similar to the one of the spherical Bessel functions. For second kind, instead, the functions have a vertical asymptote near zero that moves toward the right with increasing order. Furthermore, they have lot of zero-crosses: this creates a problem when those functions compare at the denominator.

## Inhomogeneous Wave Equation

So, up to now we have saw solutions with the very strong assumption of source-free volume. Let us introduce acoustical sources with boundary conditions. By considering a volume with source distribution, acoustic field must satisfy the **inhomogeneous wave equation**

$$\nabla^2 p(\underline{r}, t) - \frac{1}{c^2} \frac{\partial^2 p(\underline{r}, t)}{\partial t^2} = -\frac{\partial q(\underline{r}, t)}{\partial t}$$

Where  $q(\underline{r}, t)$  represents the flow per unit volume.

By applying the Fourier transform, this equation lead to the **inhomogeneous Helmholtz equation**

$$\nabla^2 P(\underline{r}, \omega) - \frac{\omega^2}{c^2} P(\underline{r}, t) = -j\omega Q(\underline{r}, t)$$

The equation is now equal to  $Q$  that is the volume velocity.

We know that if we put as input a Kronecker delta to a LTI it returns the impulse response of the system.

If we are able to extend this concept into our specific scenario, we can obtain something similar to its impulse response (or equivalently to the transfer function on frequency domain). We can generalise the definition of the impulse with a spatial-temporal impulse placed in the space-temporal point  $(\underline{r}', t')$ .

$$q(\underline{r}, t) = \delta(\underline{r} - \underline{r}')\delta(t - t')$$



Notation:  
 Kronecker delta square brackets  $\delta[n]$   
 Dirac's delta round brackets  $\delta(t)$

Practically,  $q(\underline{r}, t)$  is a point source localised in a specific point in the space-time domain that emits an impulse. If we use it as input in our differential equation, what we get is the solution of the wave equation located in  $\underline{r}'$  and delayed by the time difference of arrival. This is exactly the solution of the wave equation.

$$\nabla^2 p(\underline{r}|\underline{r}') - \frac{1}{c^2} \frac{\partial^2 p(\underline{r}|\underline{r}', t)}{\partial t^2} = -\frac{\partial[\delta(\underline{r} - \underline{r}')\delta(t - t')]}{\partial t}$$

$$g(\underline{r}|\underline{r}', t) \triangleq p(\underline{r}|\underline{r}', t)|_{q(\underline{r}, t)=\delta(\underline{r}-\underline{r}')\delta(t-t')} = \frac{1}{\|\underline{r} - \underline{r}'\|} \delta\left(t - \frac{\|\underline{r} - \underline{r}'\|}{c}\right)$$

Seeing it as a system, it is exactly the impulse response of our homogenous equation. It describes the propagation phenomena of the homogenous equation.

If we transform it, we get the so-called Green function.

$$\begin{aligned} G_{3D}(\underline{r}|\underline{r}', \omega) &\triangleq \mathcal{F}\{g(\underline{r}|\underline{r}', t)\} = \frac{1}{\|\underline{r} - \underline{r}'\|} \mathcal{F}\left\{\delta\left(t - \frac{\|\underline{r} - \underline{r}'\|}{c}\right)\right\} = \frac{1}{\|\underline{r} - \underline{r}'\|} \frac{1}{2} e^{-j2\pi f \frac{\|\underline{r} - \underline{r}'\|}{c}} = \\ &= \frac{1}{\|\underline{r} - \underline{r}'\|} \frac{1}{2} \frac{1}{2\pi} e^{j\omega \frac{\|\underline{r} - \underline{r}'\|}{c}} = \frac{e^{-j\frac{\omega}{c} \|\underline{r} - \underline{r}'\|}}{4\pi \|\underline{r} - \underline{r}'\|} \end{aligned}$$

Therefore, the Green's function in a 3D free-field can be expressed as

$$G_{3D}(\underline{r}|\underline{r}', \omega) = \frac{e^{-j\frac{\omega}{c}\|\underline{r}-\underline{r}'\|}}{4\pi\|\underline{r}-\underline{r}'\|}$$

As we can see, it depends on the distance between the source and the observation point. It is the transfer function of the homogenous equation: the basic solution that describes the propagation of a point source in space. Thus, it can be used to describe arbitrary solutions of the wave equation:

$$p(\underline{r}|\underline{r}', t) = \int_{\mathcal{D}} G_{3D}(\underline{r}|\underline{r}', \omega) D(\underline{r}', \omega) d\underline{r}' \quad \text{with } \underline{r}' \in \mathcal{D}$$

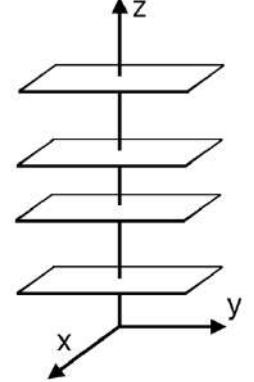
This integral equation is known as Single Layer Potential, but hey! It is nothing but the convolution for more complicate stuff.  $\mathcal{D}$  is a set that contains all the position where sources are located (even if they are continuously distributed).  $D(\underline{r}', \omega)$  is called driver signals. The product between  $D(\underline{r}', \omega)$  and  $G_{3D}(\underline{r}|\underline{r}', \omega)$  the description of the sound in space. Obviously, it can be done since linearity holds.

## Line Source

Let us see now look for the green function of a Line source. But what is a line source? A line of sources that propagate sound on plane where in a height invariant scenario (this means that sound field is invariant over all planes parallel to  $xy$  plane). How can I compute it? Simply, takning the expression of the Green's function for a source and integrate it along a line. In this setting, the exxitation function will be

$$q(\underline{r}, \omega) = \delta(x - x')\delta(y - y'), \quad z \in \mathbb{R}$$

Let us integrate it along the  $z$  axis having defined  $\underline{r}' = [x', y', z']^T$



$$G_{2D}(\underline{r}|\underline{r}', \omega) = \int_{\mathcal{D}} G_{3D}(\underline{r}|\underline{r}', \omega) dz' = \int_{-\infty}^{\infty} \frac{e^{j\frac{\omega}{c}\|\underline{r}-\underline{r}'\|}}{4\pi\|\underline{r}-\underline{r}'\|} dz' \stackrel{\text{trust it}}{=} -\frac{j}{4} H_0^{(2)}\left(\left(\frac{\omega}{c}\right)\rho\right)$$

Reasonaibly, the arument of the function depends just on  $\rho$  that is  $\rho = \sqrt{(x - x')^2 + (y - y')^2}$ .

As the Green's function  $G_{3D}(\underline{r}|\underline{r}', \omega)$  described the field measured in  $\underline{r}$  due to a point source in  $\underline{r}'$  that radiates in an omni-directional fashio,  $G_{2D}(\underline{r}|\underline{r}', \omega)$  describes the field eimeasured in  $\underline{r}$  due to a line source orthogonal to the  $xy$  plane. This resul will only depend on the distance between the measuruemnt point  $\underline{r}$  and the line source. This scenario becomes useful to simplify 3D problems where it can be assumed that the acoustic field is indiependent on the spatial coordinate.

Let us become now a little bit more practical. The sound field of a point source (spatial impulse) can be approximately generated by a closed-cabinet loudspeaker with a single driver. The approximation is valid only if we are far away enough.

The sound field of a line source (loudspeaker array, loudspeaker panel, ribbon loudspeaker, etc...) is not easy to generate. But it is possible to approximate the 2D function (sound field of a line source) with a 3D function (point source)?

Yes! By applying the large argument approximation of the Hankel function. In this way we have

$$\begin{aligned} G_{2D}(\underline{r}|\underline{r}', \omega) &= -\frac{j}{4} H_0^{(2)}\left(\left(\frac{\omega}{c}\right)\rho\right) \underset{\substack{\approx \\ \text{Large argument approx} \\ \left(\frac{\omega}{c}\right)\rho \gg 1}}{=} -\frac{j}{4} \frac{1}{\sqrt{\frac{\pi}{2}[-j\left(\frac{\omega}{c}\right)\rho]}} e^{-j\left(\frac{\omega}{c}\right)\rho} = \\ &= \frac{1}{\sqrt{16j^2 \frac{\pi}{2}[-j\left(\frac{\omega}{c}\right)\rho]}} e^{-j\left(\frac{\omega}{c}\right)\rho} = \frac{1}{\sqrt{j8\pi\left(\frac{\omega}{c}\right)\rho}} e^{-j\left(\frac{\omega}{c}\right)\rho} \end{aligned}$$

Thanks to this approximation, the function recalls the 3D function of a point source at  $\underline{r}' = [x', y', 0]^T$  observed on the  $xy$  plane, for example in a point  $\underline{r} = [x, y, 0]^T$ . It is, in fact

$$G_{3D}([x, y, 0]^T | [x', y', 0]^T, \omega) = \frac{e^{j\frac{\omega}{c}\sqrt{(x-x')^2+(y-y')^2}}}{4\pi\sqrt{(x-x')^2+(y-y')^2}} = \frac{e^{-j\frac{\omega}{c}\rho}}{4\pi\rho}$$

Let us compare the two formulas. The following equality holds as we consider two additional function to be multiplied by  $G_{3D}(\rho, \omega)$ .

$$G_{2D}(\rho, \omega) = H(\omega)A(\rho) \text{ by } G_{3D}(\rho, \omega)$$

These two “corrective” functions are:

- $H(\omega) = \sqrt{\frac{c}{j\omega}}$  called spectral shaping
- $A(\rho) = \sqrt{2\pi\rho}$  called amplitude modification

Therefore, field of a line source can be approximated on the  $xy$  plane by a point source at the root point of the line source. The approximation exhibits low pass shaping of the spectrum, compensated by the function  $H(\omega)$ . Also, the amplitude decay is different:  $\frac{1}{\sqrt{\rho}}$  for the line source,  $\frac{1}{\rho}$  for the point source. This time again we intervene with a corrective function, in particular  $A(\rho)$ .

### Boundary condition

Now that we have the equation, it is time to look for the boundary conditions. The boundary conditions can be classified as:

- Homogeneous boundary conditions that describe stationary boundaries
- Inhomogeneous boundary conditions that describe reacting boundaries

Let us start with the first boundary condition:

#### *Dirichlet boundary condition*

- Homogenous Dirichlet boundary condition (sound soft, no pressure on the boundaries (we remind that  $\partial\mathcal{V}$  describes the boundary of the volume  $\mathcal{V}$ )).

$$p(\underline{r}, \omega) = 0 \quad \forall \underline{r} \in \partial\mathcal{V}$$

- Inhomogeneous Dirichlet boundary condition (sound field on the boundary equals an arbitrary function (sound hard, rigid)).

$$p(\underline{r}, \omega) = f(\underline{r}, \omega) \quad \forall \underline{r} \in \partial\mathcal{V}$$

### **Neumann boundary condition**

- Homogenous Neumann boundary condition.

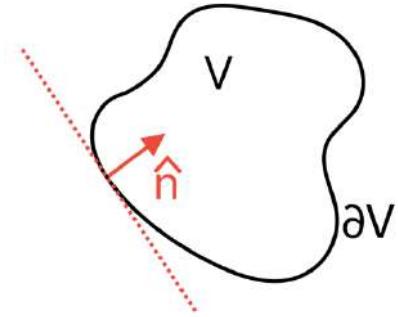
$$\langle \nabla p(\underline{r}, \omega), \hat{n}(\underline{r}) \rangle = 0 \quad \forall \underline{r} \in \partial \mathcal{V}$$

$\hat{n}$  is the normal on the boundary pointing into the domain of interest; the directional derivative of the sound field in along the normal vanishes

- Inhomogeneous Neumann boundary condition

$$\langle \nabla p(\underline{r}, \omega), \hat{n}(\underline{r}) \rangle = f(\underline{r}, \omega) \quad \forall \underline{r} \in \partial \mathcal{V}$$

The directional derivative of the sound field in the direction normal to the boundary equals an arbitrary function.  
In both cases, the condition is on the gradient of a peculiar direction.



The condition is on the gradient on a peculiar direction.

### **Sommerfeld radiation condition**

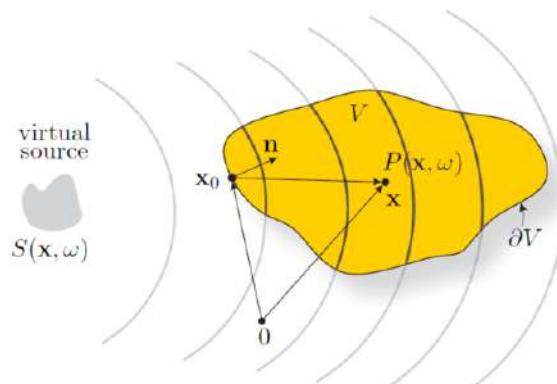
A last, more exotic condition is the so called Sommerfeld radiation condition. This condition is applied for exterior problems, therefore at infinity. No energy contribution to the sound field is originated at infinity

$$\lim_{r \rightarrow +\infty} r \left( \frac{\partial p(\underline{r}, \omega)}{\partial r} + j \frac{\omega}{c} p(\underline{r}, \omega) \right) = 0$$

### **The Kirchhoff-Helmholtz Integral**

Let us suppose to have a distribution of sources in space and a certain volume  $\mathcal{V}$  confined by a specific boundary  $\partial \mathcal{V}$ . Our aim is to reconstruct the sound field  $a(\underline{r})p(\underline{r}, \omega)$  inside the volume just relying on the acoustic field and on the boundary. We express it as the difference of two terms evaluated in each point of the boundary:

- The green function weighted on the inner product between the gradient of the pressure (acoustic field) and the unitary vector perpendicular to the tangent in the point of evaluation (normal direction towards the centre of the volume).
- The pressure that weights the inner product between the gradient of the Green function and the unitary vector perpendicular to the tangent in the point of evaluation. The spatial derivative (becoming directional due to the inner product) corresponds to a stir dipole.



Practically speaking, it is the difference between a monopole radiation and a dipole stir towards to the normal to the boundary driven by the acoustic field measured in all the points of the boundaries

Green function weighted on the directional derivative of the acoustic field

$$a(\underline{r})p(\underline{r}, \omega) = -\oint_{\partial\mathcal{V}} \left[ G(\underline{r}|\underline{r}', \omega) \langle \nabla p(\underline{r}, \omega), \hat{\underline{n}}(\underline{r}') \rangle \Big|_{\underline{r}=\underline{r}'} - p(\underline{r}', \omega) \langle \nabla G(\underline{r}|\underline{r}', \omega), \hat{\underline{n}}(\underline{r}') \rangle \right] dA(\underline{r}')$$

Discrimination term

$$a(\underline{r}) = \begin{cases} 1, & \text{for } \underline{r} \in \mathcal{V} \\ \frac{1}{2}, & \text{for } \underline{r} \in \partial\mathcal{V} \\ 0, & \text{for } \underline{r} \notin \mathcal{V} \end{cases}$$

If we translate what we have just write into the English language: the sound field in  $\mathcal{V}$  is uniquely determined by the sound pressure on  $\partial\mathcal{V}$  and by its gradient in the direction normal to  $\partial\mathcal{V}$ .

## References

- [1] P. Brémaud. Mathematical Principles of Signal Processing. Springer Science+Business Media, New York, NY, USA, 2002.
- [2] M. Vetterli, J. Kovacevic, and V. K. Goyal. Foundations of Signal Processing. Cambridge University Press, Cambridge, UK, 2014.
- [3] F. W. J. Olver, editor. NIST Handbook of Mathematical Functions. National Institute of Standards and Technology, New York, NY, USA, 2010.
- [4] N. Xiang and C. Landschoot. Bayesian inference for acoustic direction of arrival analysis using spherical harmonics. *Entropy*, 21, 2019.
- [5] E. G. Williams. Fourier Acoustics. Academic Press, London, UK, 1999.
- [6] D. Colton and R. Kress. Inverse Acoustic and Electromagnetic Scattering Theory. Springer-Verlag, Berlin Heidelberg, DE, 1992.
- [7] I. S. Gradshteyn and I. M. Ryzhik. Table of Integrals, Series, and Products. Academic Press, Burlington, MA, USA, seventh edition, 2007.
- [8] M. Abramowitz and I. A. Stegun, editors. Handbook of Mathematical Functions. National Bureau of Standards, Washington DC, USA, tenth edition, 1972.
- Brief Review of Acoustics
- [9] P. M. Morse and K. U. Ingard. Theoretical Acoustics. Princeton University Press, Princeton, NJ, USA, with errata page, first Princeton University Press edition, 1986.
- [10] L. E. Kinsler, A. R. Frey, A. B. Coppens, and J. V. Sanders. Fundamentals of Acoustics. John Wiley & Sons, New York, NY, USA, fourth edition, 2000.
- [11] P. M. Morse and H. Feshbach. Methods of Theoretical Physics, volume I. McGraw-Hill, New York, NY, USA, 1953.
- [12] S. Spors, R. Rabenstein, and J. Ahrens. The theory of wave field synthesis revisited. In Proc. AES 124th Conv., Amsterdam, NE, May 17-20 2008.

In today's lecture we want to find strategy that makes the representation lighter than the one we saw yesterday.

Firstly, we review the transformation we saw yesterday

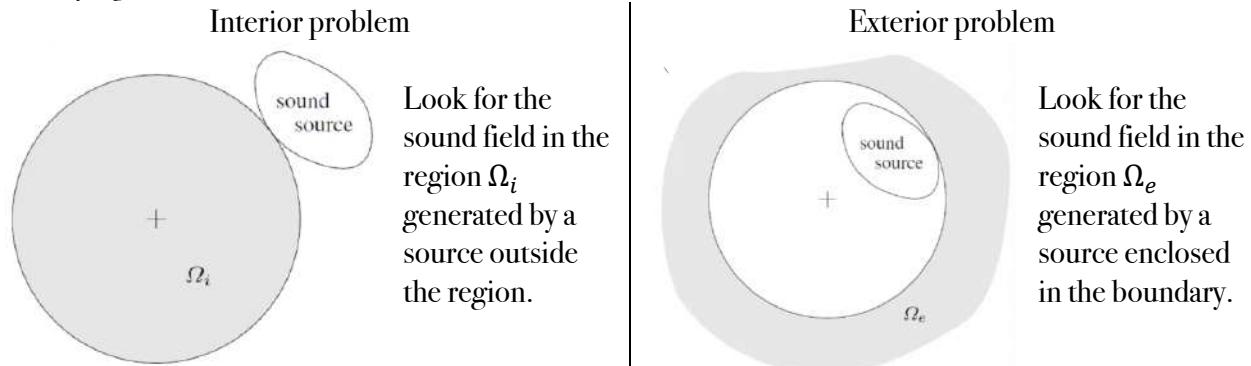
<p>Mono-dimensional domain</p> $x(t) \quad t \in \mathbb{R}$ $X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt \quad \omega \in \mathbb{R}$ $x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega)e^{j\omega t} d\omega \quad t \in \mathbb{R}$	<p>Multi-dimensional domain</p> $x(\underline{r}) \quad \underline{r} \in \mathbb{R}^D$ $X(\underline{k}) = \int_{-\infty}^{\infty} x(\underline{r})e^{-j\langle \underline{k}, \underline{r} \rangle} d\underline{r} \quad \underline{k} \in \mathbb{R}^D$ $x(\underline{r}) = \left( \frac{1}{2\pi} \right)^D \int_{-\infty}^{\infty} X(\underline{k})e^{j\langle \underline{k}, \underline{r} \rangle} d\underline{k} \quad \underline{k} \in \mathbb{R}^D$
--	--

What we want to do is to find a function as the Fourier's one able to represent a sound field. Formally, given a basis solution kind (plane wave, spherical wave, cylindrical wave) identify a set of independent variables that determine the basis solution and obtain the sound field as weighted superimposition of all possible basis solutions. Obviously, as it happens in any algebraical system, by changing the type of waves (the basis  $\beta_n(\underline{r}, \omega)$ ), also the weighting coefficients changes (the scalars,  $c_n$ ). Practically speaking, it is once again a recipe: how much of spherical wave do I have to put inside my bowl to obtain a certain kind of sound field? And how much of spherical and cylindrical wave? As we know the quantity of the ingredients, we are able to "build" the sound field. Formally, sound field is fully described by the weighting coefficients (that, since they weight the "expansion" of the sound field, are called expansion coefficients).

$$p(\underline{r}, \omega) = \sum_{n=-\infty}^{\infty} c_n \beta_n(\underline{r}, \omega)$$

Another way to see the problem is: the coefficient  $c_n$  expresses the similarities of the sound field with the basis function  $\beta_n(\underline{r}, \omega)$ .

Before addressing to how define or compute those coefficients, we have to give some useful definition on source-free volume that, we remind, is a portion of space in which there is not any sound source. Since we are defining a boundary, the space will be divided in what is inside this boundary and what is outside of it. This leads us to have two different scenarios when we want to solve an acoustic problem. Well, one step at a time. What does "to solve an acoustic problem" means? Try to estimate the sound field at a given point. We were saying that we can face two different scenarios:

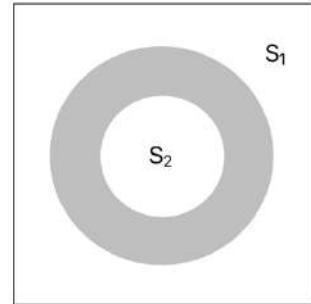


For example, the Kirchhoff-Helmholtz Integral aims to solve an interior problem: we are trying to reconstruct the sound field inside the region generated by sources that are outside the region.

What it can happen are also limit cases:

- Half space with all sources located in the other half space
- Mixture of the interior and the exterior (for instance, we have a spherical shell with one source inside the smaller radius sphere and one source outside the larger radius sphere and we are interested to retrieve the sound field in the grey region).

Let us see how to represent the different types of waves



### Plane wave representations

The first kind of solution we discussed yesterday is the so-called plane-wave. We saw two different types of plane waves: propagating plane waves and evanescent plane waves. In the second case, there is a direction along which they suffer an exponential decay. Anyway, we can represent a sound field as sums of weighted plane waves. Two different kinds of plane waves representation exists:

1. **Whittaker** representation

Derived from homogeneous wave equation, based on only propagating plane waves.

2. **Weyl** representation

Derived from the inhomogeneous wave (much complex), based on both propagating and evanescent plane waves.

We will consider just the first type. Thus, the basis solution of homogenous Helmholtz equation in Cartesian coordinates is

$$p(\underline{r}, \omega) = e^{j\langle \underline{r} \cdot \underline{k} \rangle} \quad \underline{k} \in \mathbb{R}^3$$

Our acoustic field can be thought as an inverse multidimensional Fourier transform, where  $\underline{k}$  plays the role of a spatial frequency

$$p(\underline{r}, \omega) = \left( \frac{1}{2\pi} \right)^3 \iiint_{\mathcal{D}} A(\underline{k}) e^{j\langle \underline{k} \cdot \underline{r} \rangle} d^3 \underline{k} \quad \text{where } \mathcal{D} = \left\{ \underline{k} \in \mathbb{R}^3 \mid \|\underline{k}\| = \frac{\omega}{c} \right\}$$

The coefficients  $A(\underline{k})$  depends on our wave number vector and of course multiply our basis.

It is generally a complex number; it means that encodes the amplitude and phase of each plane wave contribution to the integral. It is integrated for different propagation directions.

There is also an alternative representation.

$$p(\underline{r}, \omega) = \left( \frac{1}{2\pi} \right)^3 \iint_{\mathcal{S}} A(\theta, \phi, \omega) e^{j\frac{\omega}{c} \langle \hat{\underline{k}}, \underline{r} \rangle} \sin(\theta) d\theta d\phi \quad \text{where } \mathcal{S} = \{(\theta, \phi) \in [0, \pi] \times [0, 2\pi]\}$$

Here, we can notice is the fact that this expression comes from a simple change of variables.

$$\underline{k} = \left( \frac{\omega}{c} \right) \hat{\underline{k}}, \quad \|\hat{\underline{k}}\| = 1, \quad \hat{\underline{k}} = [\sin(\theta) \cos(\phi), \sin(\theta) \sin(\phi), \cos(\theta)]^T$$

In this case the coefficient  $A(\theta, \phi, \omega)$  depends again on the direction but in a spherical coordinate system. The contribution is summed along all the possible integration angles.

Why is this particularly useful? Because we can think of a sound field as a superposition of planewaves where the coefficients encode magnitude and phase. These coefficients are known as Herglotz density.

## Spherical Harmonics Representations

This time again, we assume a priori that the solution can be written as a product of functions that depends on a single coordinate. This is particularly useful for their properties we saw yesterday.

The last property of spherical harmonics is their unitary. We can think of the basic solution in this way: a composition of spherical Bessel's function and Henkel's function of the second type.

$$p(\underline{r}, \omega) = R(r)\Theta(\theta)\Phi(\phi) = \underbrace{R(r)Y_l^m(\theta, \phi)}_{\text{spherical harmonic waves}}$$

Where, we remind,  $R(r)$  can be written

- in terms of Spherical Bessel functions

$$R(r) = R_1 j_l\left(\left(\frac{\omega}{c}\right)r\right) + R_2 y_l\left(\left(\frac{\omega}{c}\right)r\right)$$

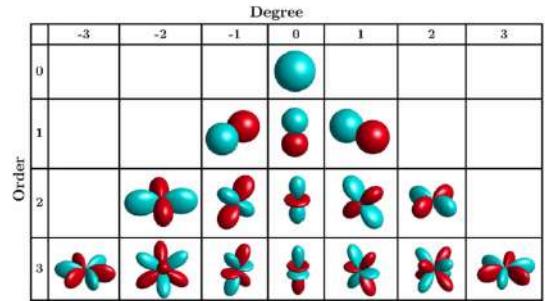
- in terms of Spherical Hankel functions

$$R(r) = R_3 h_l^{(1)}\left(\left(\frac{\omega}{c}\right)r\right) + R_4 h_l^{(2)}\left(\left(\frac{\omega}{c}\right)r\right)$$

Where:

$$h_l^{(1)}\left(\left(\frac{\omega}{c}\right)r\right) = j_l\left(\left(\frac{\omega}{c}\right)r\right) + jy_l\left(\left(\frac{\omega}{c}\right)r\right)$$

$$h_l^{(2)}\left(\left(\frac{\omega}{c}\right)r\right) = j_l\left(\left(\frac{\omega}{c}\right)r\right) - jy_l\left(\left(\frac{\omega}{c}\right)r\right)$$



Let us have the chance to look again to these nice plots.

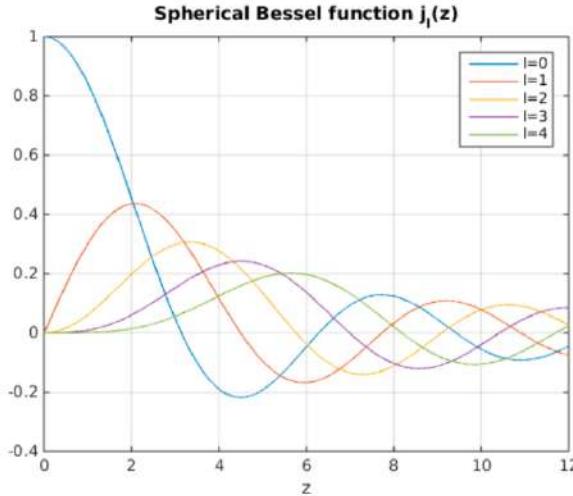
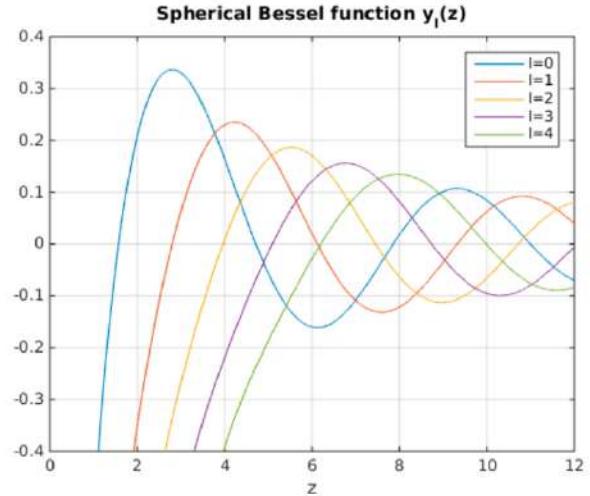
Following similar principle as this one, we can define an acoustic field as an expansion of spherical harmonic waves

$$\begin{aligned} p(\underline{r}, \omega) &= \sum_{l=0}^{\infty} \sum_{m=-l}^l \left( A_{lm}(\omega) h_l^{(1)}\left(\left(\frac{\omega}{c}\right)r\right) + B_{lm}(\omega) h_l^{(2)}\left(\left(\frac{\omega}{c}\right)r\right) \right) Y_l^m(\theta, \phi) = \\ &= \sum_{l=0}^{\infty} \sum_{m=-l}^l \left( C_{lm}(\omega) j_l\left(\left(\frac{\omega}{c}\right)r\right) + D_{lm}(\omega) y_l\left(\left(\frac{\omega}{c}\right)r\right) \right) Y_l^m(\theta, \phi) \end{aligned}$$

We are able to completely characterise the sound field if we know the set of coefficients  $A_{lm}, B_{lm}$  or the set of coefficients  $C_{lm}, D_{lm}$ .

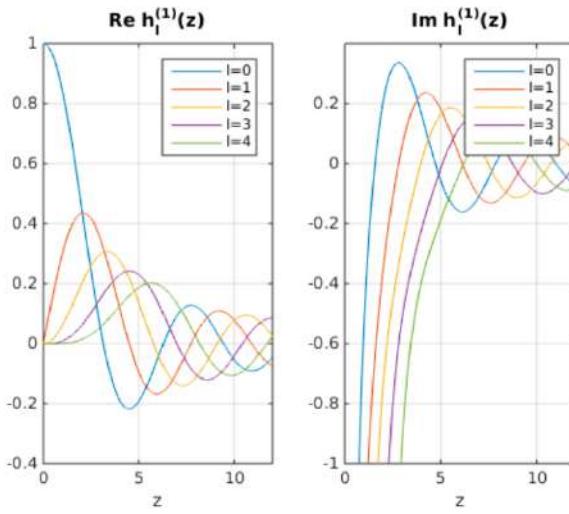
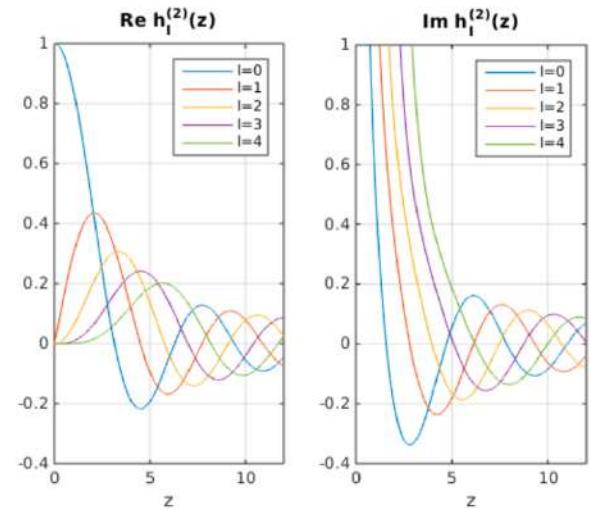
Depending on the problem that we are trying to solve, we can change the sets of coefficients. We will discuss this point a little bit over.

Let us see the plot of the Bessel's function with the concept of interior and exterior problem in mind.

(a)  $j_l(z)$ (b)  $y_l(z)$ 

The two functions depend on the value on  $r$  (radial distance from the origin of our reference system). The only function that starts from 1 is the one with order 0. The intuition is in the fact that the argument depends on  $\frac{\omega}{c}r$ : if we are trying to solve acoustic problems which involves small values of  $r$ , spherical Bessel's function of the second type cannot be used! As we approach zero, the function goes to  $-\infty$ . Therefore, for interior problem, this kind of functions must be neglected as they generate numerical problems. On the other hand, spherical Bessel functions of first type will be particularly desirable when  $r \rightarrow 0$ . So, the recipe is made in a way that the contributions are properly pondered in order to avoid numerical problems.

Similar consideration can be applied to spherical Hankel's function. Again, the behaviour is more or less similar.

(a)  $h_l^{(1)}(z)$ (b)  $h_l^{(2)}(z)$ 

In conclusion, due to the properties of spherical Bessel and spherical Hankel functions, the following choices are widely adopted:

- Spherical Bessel functions of the first kind,  $j_l\left(\left(\frac{\omega}{c}\right)r\right)$ , are used to represent incoming acoustic fields, i.e. acoustic fields due to sources far from the origin and observed around the origin (as in the interior problem).

- Spherical Hankel functions, i.e.,  $h_l^{(1)}\left(\left(\frac{\omega}{c}\right)r\right)$  or  $h_l^{(2)}\left(\left(\frac{\omega}{c}\right)r\right)$ , are used to represent outgoing acoustic fields, i.e. acoustic fields due to sources close to the origin and observed in an external region (as in the exterior problem). In fact, as we noticed, when  $z \rightarrow 0$  we have that  

$$|h_l^{(2)}(z)| \rightarrow \infty$$

Let us try now to retrieve those coefficients for an internal scenario. The sound field can be represented by the following synthesis equation called inverse spherical harmonics expansion.

$$p(\underline{r}, \omega) = \sum_{l=0}^{\infty} \sum_{m=-l}^l C_{lm}(\omega) j_l\left(\left(\frac{\omega}{c}\right)r\right) Y_l^m(\theta, \phi)$$

In order to find the coefficients, we have to do nothing but apply the analysis equation called spherical harmonics expansion

$$C_{lm}(\omega) = \frac{1}{j_l\left(\left(\frac{\omega}{c}\right)r\right)} \int_0^{2\pi} \int_0^\pi p(\underline{r}, \omega) Y_l^{-m}(\theta, \phi) \sin(\theta) d\theta d\phi$$

As we notice as we are solving the inverse problem (trying to estimate the coefficients to weights the spherical harmonics Bessel functions) we play with the argument of the functions to have coefficients that are not in prohibited points of the functions.

The same can be done with external sound fields (exterior problem)

- Inverse Spherical harmonics expansion (synthesis)

$$p(\underline{r}, \omega) = \sum_{l=0}^{\infty} \sum_{m=-l}^l B_{lm}(\omega) h_l^{(2)}\left(\left(\frac{\omega}{c}\right)r\right) Y_l^m(\theta, \phi)$$

- Spherical harmonics expansion (analysis)

$$B_{lm}(\omega) = \frac{1}{h_l^{(2)}\left(\left(\frac{\omega}{c}\right)r\right)} \int_0^{2\pi} \int_0^\pi p(\underline{r}, \omega) Y_l^{-m}(\theta, \phi) \sin(\theta) d\theta d\phi$$

Since we cannot go up to infinity, what we practically do is truncating the series and approximate it with a  $L$ -elements sum.

$$p(\underline{r}, \omega) \approx \sum_{l=0}^{L-1} \sum_{m=-l}^l B_{lm}(\omega) h_l^{(2)}\left(\left(\frac{\omega}{c}\right)r\right) Y_l^m(\theta, \phi)$$

Therefore, the sound field can be described by  $L^2$  coefficients.

Let us recap all the consideration we have done yesterday about Bessel functions  $j_l(\cdot)$ :

- the higher the order  $l$ , the higher is the argument  $\left(\frac{\omega}{c}\right)r$  at which the maximum occurs
- low orders describe well the sound field close to the origin
- high orders describe the sound field far from the origin

In our truncation operation we need a rule of thumb to consider how much we can approximate the infinite series with a finite one. We say that the bandlimited expansion is valid if

$$\frac{\omega}{c}r_{L-1} < (L - 1)$$

Where  $r_{L-1}$  is the radius of validity of the expansion, inversely-proportional with temporal frequency  $\omega$ .

Let us now consider the effect of this approximation. As we can see from the picture on the right, by decreasing  $L$ , the image becomes more “spherical” with more circles instead of straight lines.

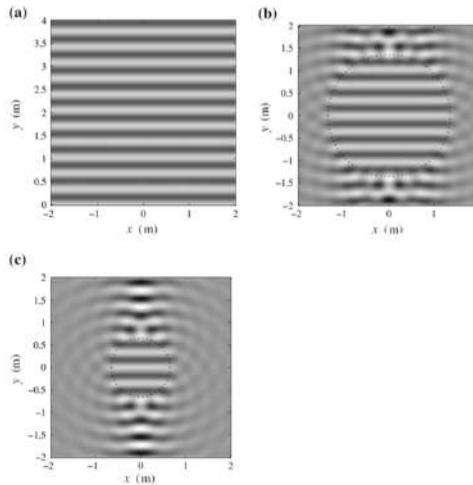


Figure: Effects of bandlimited expansion. Ideal field (a). Bandlimited fields:  $L = 25$  (b) and  $L = 12$  (c). From [1, Fig. 2.7].

### Cylindrical Harmonics representations

What it misses is a brief discussion on cylindrical coordinates. These are summations of basis functions and there are several scenario in which cylindrical coordinates can become useful. We report just the synthesis equation

$$p(r, \omega) = \sum_{m=-\infty}^{\infty} \left( \frac{1}{2\pi} \int_{-\infty}^{\infty} c_m(\omega, k_z) J_m(k_\rho \rho) e^{jk_z z} dk_z \right) e^{jm\phi}$$

The equation is valid in a cylindrical region around the  $z$  axis with a radius  $\rho$  smaller than the distance of the nearest source.

### References

- [1] J. Ahrens. Analytic Methods of Sound Field Synthesis. Springer-Verlag, Berlin, DE, 2012.
- [2] E. G. Williams. Fourier Acoustics. Academic Press, London, UK, 1999.
- [3] D. Colton and R. Kress. Inverse Acoustic and Electromagnetic Scattering Theory. Springer-Verlag, Berlin Heidelberg, DE, 1992.
- [4] F. W. J. Olver, editor. NIST Handbook of Mathematical Functions. National Institute of Standards and Technology, New York, NY, USA, 2010.
- [5] N. Xiang and C. Landschoot. Bayesian inference for acoustic direction of arrival analysis using spherical harmonics. Entropy, 21, 2019.
- [6] R. A. Kennedy, P. Sadeghi, T. D. Abhayapala, and H. M. Jones. Intrinsic limits of dimensionality and richness in random multipath fields. IEEE Transactions on Signal Processing, 55(6):2542–2556, June 2007.

## Spatial Perception of Auditory Events

In our case we can define the sound localisation as the ability to identify the spatial position (direction and distance) of the origin of a sound stimulus. The cues that the auditory systems exploits to solve the sound localisation problem are:

- Interaural differences
- Spectral information
- Correlation
- Pattern Matching

If we are interested into design a sound localisation experiments, we are then interested in measuring the capability of a subject to understand the position. A problem like this one depends on several factors:

- Choice of the stimulus
- Acoustic conditions,
- Etc...

Also, the localisation error has to take into account how a subject makes errors when hears sounds.

Therefore, the localisation errors will be a combination of:

1. Errors due to the measurement setup;
2. Errors actually associated with the perception.

A naïve solution to our problem would be given by the following scenario:

- Distribute a number of fixed loudspeakers in space and switch the sound stimulus between them.
- The listener indicates the loudspeakers perceived as the one generating the sound.

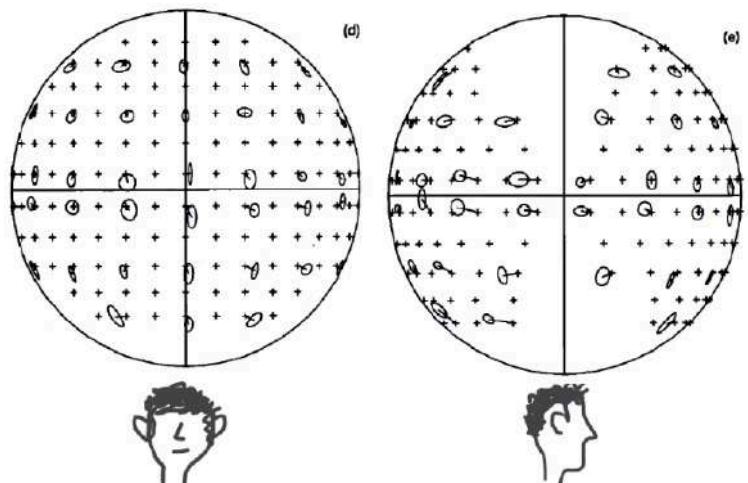
Unfortunately, we bump into a problem: restricting the number and location of targets constrains the listener's response. This is because the listener is biased from many factors, for example it can see the speakers. In other words, this means that the brain merges the information that comes from eyes to localise the loudspeaker.

A more accurate solution could be using a hidden (that cannot be seen) and movable sound source. A 3D tracking device is mounted on the listener's head to enable "head pointing". The listener points its nose towards the perceived target direction as turning the head towards the source is an instinctive natural behaviour (motivated by the need of including the sound source in the potential visual area).

If we perform such an experiment with this addition, we notice that subjects are not good in understand emispherically where the source is located. These are, in fact, the results:

- Perceived location is about  $20^\circ$  off the actual location.
- azimuthal localization is correct, but the hemisphere is wrong
- easy to extract from listener's responses to deal with them separately

The plot on the right, reports the results.



## Primary Auditory Cues – Directional Cues in free-field

In today's lecture we will resume the part we saw on Friday. We were discussing the primary auditory cues, so those tips that the brain exploit to understand from where a sound is coming. The first class of primary auditory cues are the so-called interaural cues:

- **Interaural level difference**

It represents the difference in amplitude from the sound we hear from one ear to the other. The closer will be the ear to the source, the higher will be the sound. Furthermore, it takes into account the acoustical attenuation of the sound by the head due to absorption and reflections. It depends on the dimension of the head: the more the wavelength is small, the higher will be the effect of shadowing of our head.

ILD varies with azimuthal angle:

- sound from sources located at one side is maximally attenuated at the far ear
- sound from front sources arrives at both ears with similar intensity.

ILD varies with frequency (relationship between head size and wavelength)

- from less than 2dB at 200 Hz
- to over 20dB above 6 kHz

Taking into account the frequency effects, ILD does not exhibit a simple relationship to azimuth in real listeners. At the end, we can say that ILD is relevant for our task only for mid-high frequencies (above  $\sim 700$  Hz).

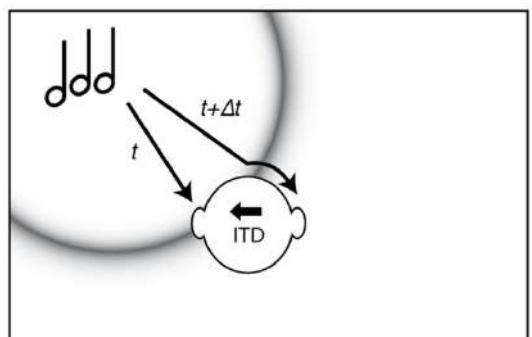
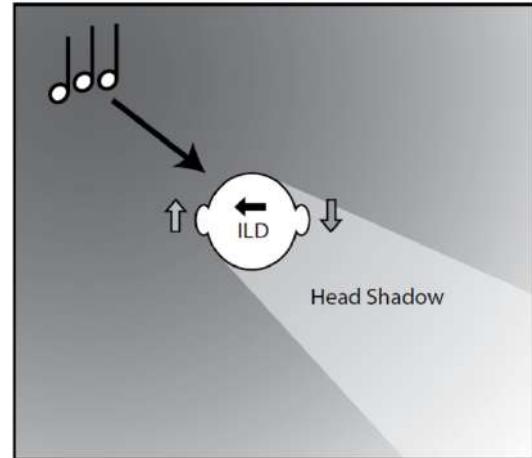
- **Interaural time difference**

Difference in time of a sound wave from an ear to the other, in this range of frequency the head shadowing effect is more relevant.

ITD arises from different distances travelled by sound in reaching near and far ears

- frontal sound sources have ITD  $\sim 0 \mu s$
- lateral sound sources have maximum ITD  $\sim 600 \mu s$

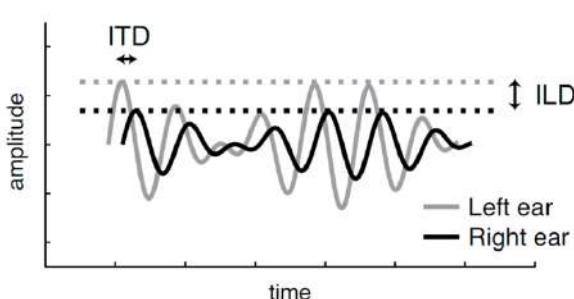
ITD is relevant for low frequencies (below  $\sim 1.5$  kHz), that is why we are much more sensible at signal phases at lower frequency.



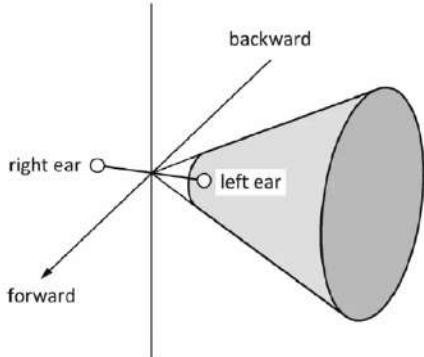
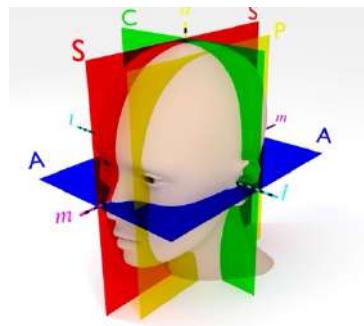
## Duplex Theory

By combining the two indicators we can obtain a new theory called **duplex theory**: this is one of the first

psychoacoustic theory presented at all. In the plot on the left, the difference in amplitude is the ILD while the difference in time is the ITD. With those two primary cues. Everything seems very nice. But this the model is too much simple, with lot of limitations! If both the quantities are zero, how can we say if the source is coming from the front or from behind?



We have to generalise the problem: let us consider the plane (median plane) longitudinal to our head perpendicular to the interaural axis (the red one). And let us now consider sources that lie on this plane: they all will have same ILD and same ITD. We are confused, if two sounds come from two sources that lies on a plane on which ILD and ITD is the same, we are not able to link the two sounds to the relative sources. We can see this case of the plane as degeneration of the cone. We call these cones "cone of confusion". The duplex theory is not power enough.



All the sounds coming from sources that lie on a point of a cone of confusion will have the same ITD and the same ILD.

We need additional cues on which to rely. Just relying on these two cues we have some cones of confusion that does not allow us to pinpoint the sources.

Furthermore, ITD and ILD have some limitations!

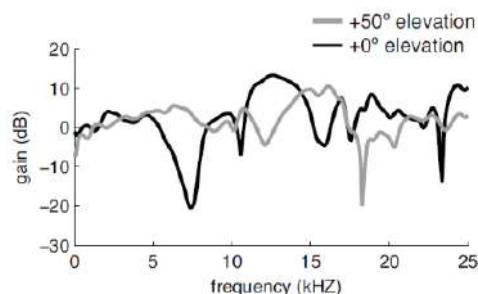
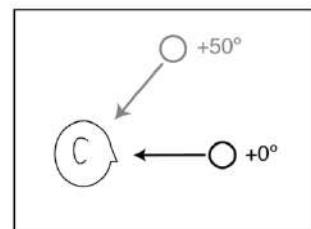
	<b>ITD JND</b>	<b>ILD JND</b>
Noise	$6 \div 10 \mu s$	$0.5 \div 1 dB$
Clicks	$20 \div 50 \mu s$	$1 \div 2 dB$
Tone 500 Hz	$> 25 \mu s$	$0.5 \div 1 dB$
Tone 4 kHz		$0.5 \div 1 dB$

How can we disambiguate cones of confusions? One of the possible strategies to do so is passing from a static scenario to one in which we are allowed to move our head: head rotation in azimuthal plane alters ITD and ILD in opposite ways for front and back sources Why so? When we rotate our head, we intercept cone of confusion one with the other with different interceptions angles.

These head movements are really useful when we are in scenario in which sounds last long. But with short sounds, the head movements are not enough to disambiguate a cone of confusion.

### *Monoaural Spectral cues*

We have to exploit other cues in order to do it. We introduce the monoaural spectral cues. If we measure the difference at the ear canal with sounds that are coming from different angles, we have different spectral components. This depends on the shape of the pinna. So, each one of us perceive the pinna-filtered signal in a peculiar way.



Head:	1.5 KHz
Pinna:	3 KHz

The effect is not easy to characterize

- rear at 1 kHz
- front from 2 to 4 kHz
- overhead at 8 kHz
- rear at 12 kHz

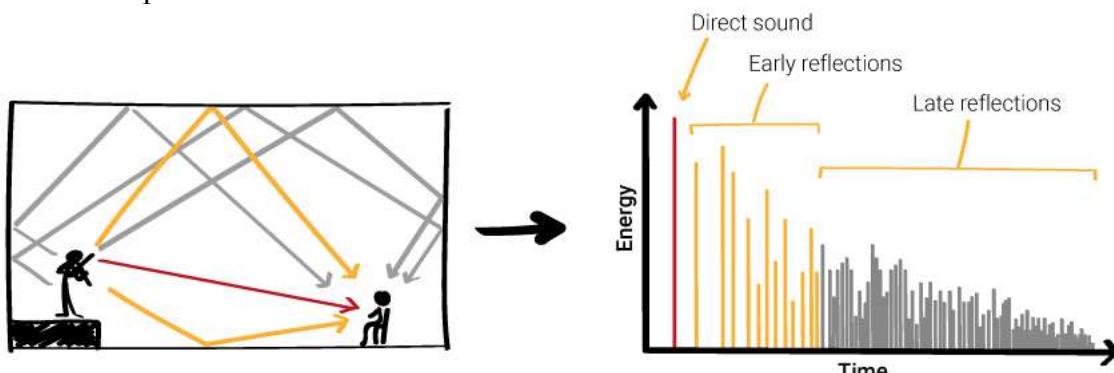
Similar behaviour is observed when sound spectrum at the eardrum is recorded, due to sources from various elevations: spectrum is altered by head, torso, pinna, etc.

### Primary Auditory Cues – Directional Cues in reverberant environments

Since it is very difficult to describe this effect, we need to think to rely on something different. Instead of considering just the pure signals, let us see what happens when we reintroduce reverberations.

While in free-field interaural and spectral cues are static, in reverberant environments cues evolve with the duration of the sound. In fact, echoes and reverberation sum to the direct sound and cues carried by the onset may be different from cues carried by the late part.

The impulse response of a reverberation room has a first spike and then others that intensifies in density as they decrease in amplitude due to reflections.



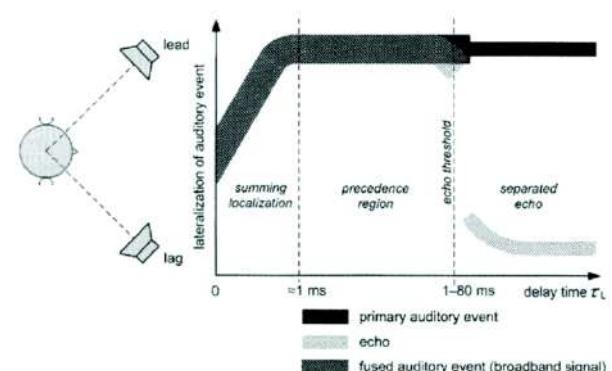
This dynamicity (the fact that the impulse response changes according to the reflections and that they change according with the angle) helps us to identify the position. Sensitivity to cues evolution depends strongly on the stimulus.

### Precedence effect

Let us now discuss another effect related to the localisation of a sources in a 3D environment: the precedence effect. In this case localisation is not affected by early reflections, direction is determined by the first wavefront. This basic mechanism allows a man to understand perception in reverberant environments. These effects stand on the basis for stereo panning techniques.

Let us consider this scenario, we have a listener that is exactly between two speakers in a perfectly symmetrical situation. The first speaker is the lead (the one that produces the sound first), then we have the lag (a speaker that replicates the sound after a certain time).

In a first region called summing localisation region, the time lag allows us to exploit the delay to perceive the position of the source. Then, in the second region we perceive just the lead emitted sound. (this is why the direction is identified by the first waveform that arrives to our ear). If we increase more the delay, we perceive an echo.

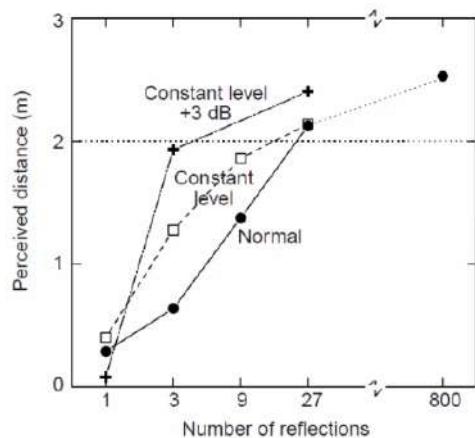
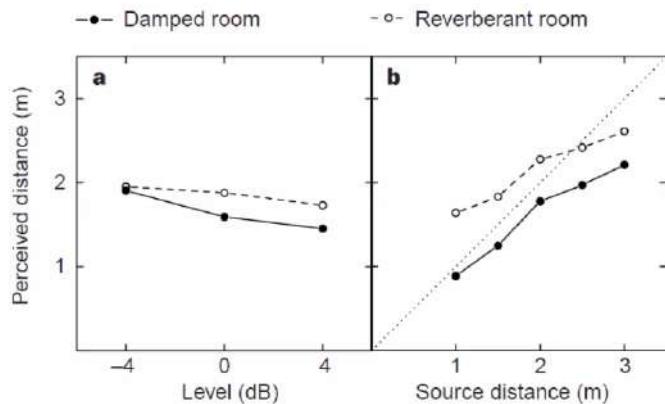


## Distance estimation

We are much better able to solve a DOA problem instead of a distance perception one, especially in free space. The human behaviour shows an inability to accurately evaluate distance of sound sources in the free field. This is because of the fact that the only distance cue is loudness, which is confused with the level of sound itself. The task becomes a bit easier in enclosures since, in that case, distance is evaluated through the energy ratio of direct and reverberant sound. This helps us since the reverberation has also a decorrelation effect: different directions (for the two ears) produce different filtered signal making them decorrelated. Another cue is provided by the timbral differences between direct and reverberant portions of the sound. Let us consider two sources located at two meters to understand if this cues really allows us to understand how much a source is far from us.

The result of the experiment reported on the right. What we observe is that the perception in a damped room is underestimated while in the opposite case there is an overestimation of the distance.

Furthermore, we notice that the perception of distance increases with the number of reflections



In the graph on the left, constant level stands for the fact that the reflections are artificial and maintained constants.

## Head-Related Transfer Function

So, we have wrapped up all the consideration related to the primary cues. Let us now address to a more overall model that could describe the auditory system.

As we saw, primary auditory cues (ITD, ILD, etc.) provide a useful functional modelling of the auditory system: they allow to predict sound location in simple scenarios, but they fail in complex scenarios.

At listening sound pressure levels, acoustic processes can be regarded as linear and time-invariant.

Considering that, linear system theory can be applied to describe them where the effect of listener's body, hairs, pinnae, etc. can be regarded to as a transfer function from a sound source and the ear canal.

So, how can we do it? With the so-called head-related transfer function (HRTF).

The HRTF is the transfer function of the acoustical system composed by a far-field sound source, the listener body and a sound receiver (a microphone) placed at the entrance of the ear canal close to the ear drum (tympan).

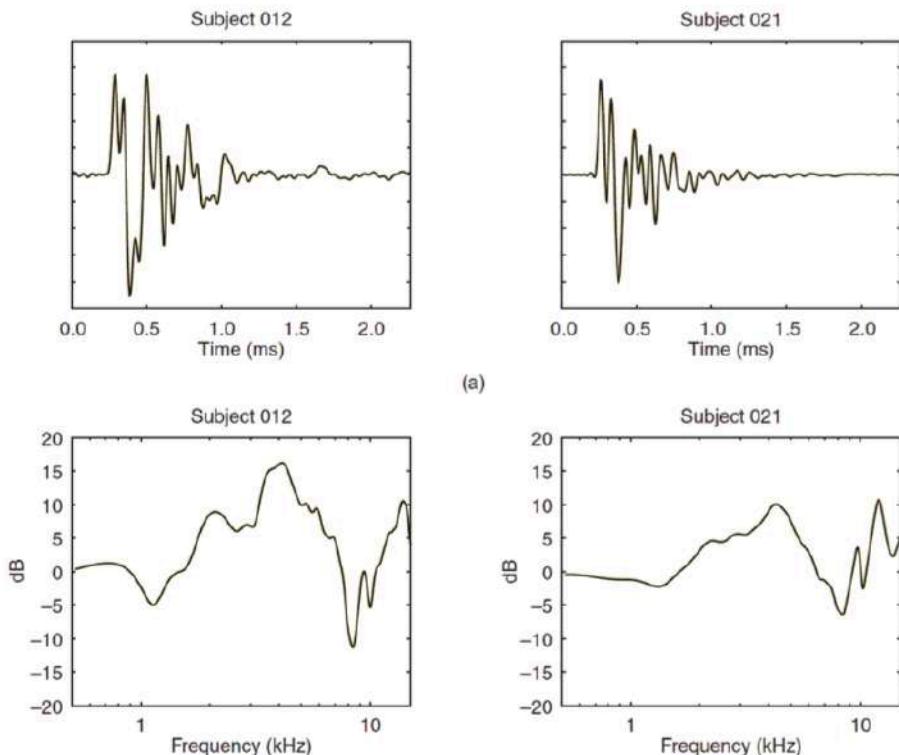
It is defined as

$$\frac{\mathcal{F}\{\text{Sound pressure at ear}\}}{\mathcal{F}\{\text{sound pressure reference}\}}$$

Although there are different waves to define this sound pressure, generally is done by taking the reference pressure in far-field with a source at the centre of the listener's head with the listener absent.

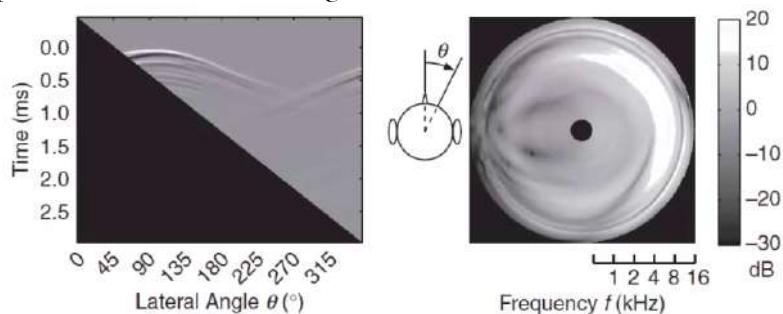
HRTF is independent on test range (distance of the sound source) only if the source is in the far field. The inverse Fourier transform of the HRTF is the HRIR that stands for head related impulse response.

Let us consider now sounds coming from the same direction and are perceived by different listeners.

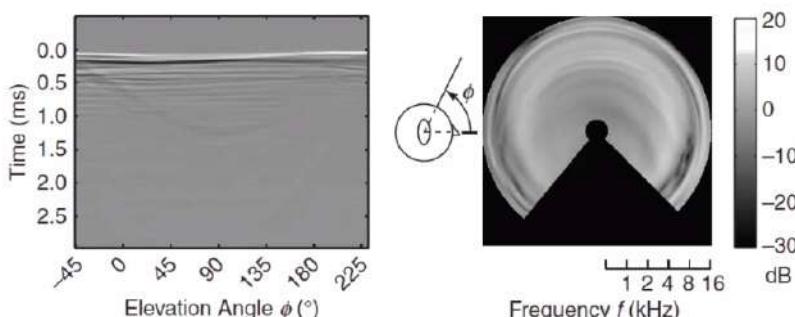


The notch we have in the frequency graph could be related to the head since the head region effects is around that frequency. The differences in notches and resonances are due to the fact that different people have different auricles.

The two functions depend also on the azimuth angle (that in this context is called  $\theta$ )



For  $0^\circ$  the source is located directly in front of the listener.



The black triangle tells us that we are not considering angles that are related to our shoulders.

How can we measure the transfer functions? There are several ways:

1. Microphone into subject's ears
2. Play a stimulus with known spectrum (pseudo-random noise, Golay sequences, sine sweeps) through loudspeaker placed at specific azimuth and elevation
3. Perform system identification (estimate the impulse response)
  - a. identify TF due to measurement apparatus (loudspeaker, microphone, etc.)
  - b. remove these components

Let us define now the data model

Firstly, the notation

- Sound stimulus  $s(t)$
- Apparatus IR  $c(t)$
- HRIR  $h_l(t)$  and  $h_r(t)$
- Recorded signals  $m_l(t)$  and  $m_r(t)$

The Microphone signals, can be thus expressed as

$$\begin{aligned} m_l(t) &= s(t) * c(t) * h_l(t) & \mathcal{F}\{m_l(t)\} = M_l(\omega) &= S(\omega)C(\omega)H_l(\omega) \\ m_r(t) &= s(t) * c(t) * h_r(t) & \mathcal{F}\{m_r(t)\} = M_r(\omega) &= S(\omega)C(\omega)H_r(\omega) \end{aligned}$$

Let us suppose to have experimentally measured (without listener) the transfer function of the apparatus. More precisely, the modulus of the transfer function of the apparatus  $|C(\omega)|$ .

From that we can retrieve the HRTF magnitude of both the two ears

$$|H_l(\omega)| = \frac{|M_l(\omega)|}{|S(\omega)||C(\omega)|} \quad |H_r(\omega)| = \frac{|M_r(\omega)|}{|S(\omega)||C(\omega)|}$$

The same can be done supposing to know the phase of the transfer function of the apparatus  $\angle C(\omega)$

$$\angle H_l(\omega) = \angle M_l(\omega) - \angle S(\omega) - \angle C(\omega)$$

$$\angle H_r(\omega) = \angle M_r(\omega) - \angle S(\omega) - \angle C(\omega)$$

Ready to write the two functions:

- HRTR

$$H_l(\omega) = |H_l(\omega)|e^{j\angle H_l(\omega)} \quad H_r(\omega) = |H_r(\omega)|e^{j\angle H_r(\omega)}$$

- HRIR

$$h_l(t) = \mathcal{F}^{-1}\{H_l(\omega)\} \quad h_r(t) = \mathcal{F}^{-1}\{H_r(\omega)\}$$

Note: a more robust estimation of HRTF is based on a minimum phase model of the transfer function combined with an all-pass filter dependent on an estimated ITD.

What is often done is to rely on a database. We report some of the most famous:

- The CIPIC HRTF database
  - Developed at U. C. Davies and published in 2001

- 45 subjects (2 KEMAR mannequins – small pinna and large pinna –, 27 men and 16 women)
- HRTF at 1250 directions for each ear of each subject
- Includes some anthropometric measures

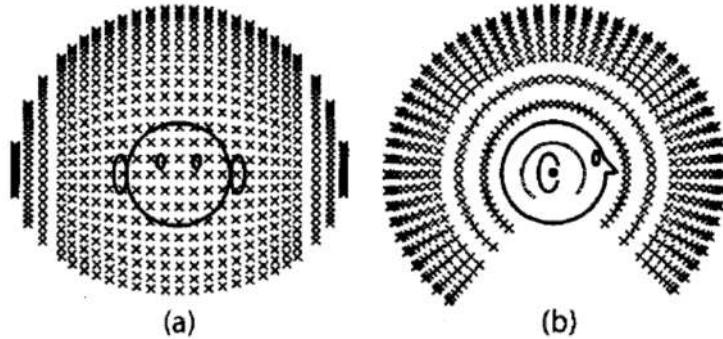


Figure: Locations of data points in CIPIC database. From [8, Fig. 1].

- Subject seated at centre of 1 m radius hoop (unconstrained, but monitoring of subject's head position).
- Loudspeakers mounted at various positions along the hoop.
- Test signals are Golay sequences,  $F_s = 44.1 \text{ kHz}$ , resolution 16 bit
- Clinic probe microphones inserted into subject's ear canal
- Recorded signals windowed to remove room reflections
- Length of each HRIR is 4.5 ms (i.e. 200 samples)



Figure: CIPIC probe microphones.

- Mathematical models
  - based on principal components (PCA, SVD, KL)
  - based on spherical harmonics

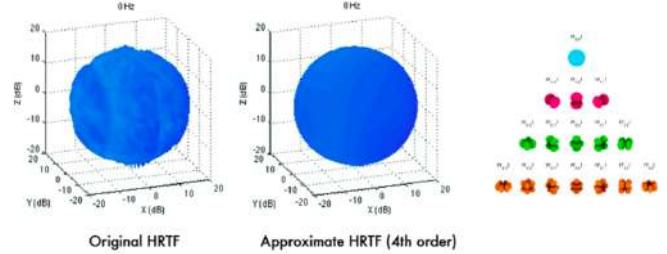


Figure: Approximation of HRTF with spherical harmonics.

- Numerical models
  - based on principal components (PCA, SVD, KL)
  - based on spherical harmonics

- The SYMARE database
  - Joint project of University of Sidney and University of York
  - 61 subjects



Figure: SYMARE population. From [14, Fig. 1].

- Acoustic HRIRs
  - measurements in anechoic chamber
  - audio microphones (Sennheiser KE 4-211-2) at the entrance of the ear's canal
- MRI data
  - 2 MRI scans for each subject (870 µm and 3 mm resolutions)
  - from top head to 15 cm below shoulder line
- Generate surface meshes from MRI data



Figure: MRI Scanner Philips 3T Achieva MRI.



Figure: Surface meshes around right ear for 6 subjects. From [14, Fig. 2].

- Fast Multipole Boundary Element Method to simulate HRIRs from surface meshes (use acoustic reciprocity principle)
  - place a source on a mesh element corresponding to the ear's canal
  - simulate propagation from that point to a uniform grid of receivers placed on a sphere around the subject's head.

## Open Research topics

Either measurement of the HRIR and scanning the MRI require prohibitive costs for consumers. Polimi has developed an algorithm that returns preliminary results with LeapMotion and Kinect (and less than 300 euro). Now this system is adopted by Apple.

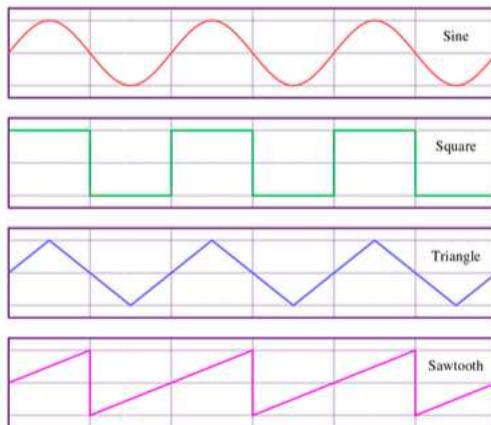
## Signal Modelling: Signal based approach (Part 1)

Let us now address to the problem of synthesis. In order to do sound synthesis, there are two approaches:

1. Signal based approach: manipulating a signal;
2. Model based approach: model the system that produces a specific sound. The complexity is embedded inside the model that produces the signal.

Let us start with the first one: we have to tackle with the most fundamental problem of signal generation. What is an oscillator?

The oscillator is a mechanism for generating a repeating waveform with a specified fundamental frequency and amplitude. It characterised by the waveform shape. We report the four most famous waveforms.



This is the fundamental building-block of any synthetiser. Firstly, we will focus on sinusoidal oscillators: how can we generate a sinusoidal oscillator? By simply using a mathematical library that implements transcendental functions.

$$y(n) = A \sin(\theta(n))$$

With the following phase wrapping (if needed)

$$\theta(n) = \begin{cases} \theta(n-1) + \Delta\theta, & \text{if } \theta \leq 2\pi \\ \theta(n-1) + \Delta\theta - 2\pi, & \text{if } \theta > 2\pi \end{cases} \quad \text{where } \Delta\theta = \frac{2\pi f_0}{F_s}$$

And if the mathematical library is not available? We have to rely on iterative generation mechanism, in particular on recursive discrete time systems.

The problem of dealing with oscillators has firstly come up at the beginning of the 20<sup>th</sup> century when engineers had to solve problems with the stability of feedback in amplifiers. The feedback is nothing but a delay line. Let us start by considering the well-known trigonometric equation

$$\cos(\phi + \theta) = 2 \cos(\theta) \cos(\phi) - \cos(\phi - \theta)$$

We notice that the formula is a recursive one: the next value of the cosine can be thought of

$$\frac{\text{next sample}}{\text{current sample}} = 2 \cdot \cos(\theta) - \frac{\text{previous sample}}{\text{current sample}}$$

The output at the next sample depends on the previous two values.

We can consider also these two coupled (coupled since one is defined by the other) form

$$\cos(\phi + \theta) = \cos(\phi) \cos(\theta) - \sin(\phi) \sin(\theta)$$

$$\sin(\phi + \theta) = \cos(\phi) \sin(\theta) + \sin(\phi) \cos(\theta)$$

Each equation uses its past values and also past values of the other equation. From a system like that, the oscillator output frequency is  $f_0 = \frac{\theta F_s}{2\pi}$

Let us define the data model.

From the first equation we take  $x_1 = \cos(\phi)$  and  $x_2 = \sin(\phi)$  and we call them state variables. These two variables will represent the past values.

For the new values, we define  $\hat{x}_1 = \cos(\phi + \theta)$  and  $\hat{x}_2 = \sin(\phi + \theta)$ . As we can notice, new state variables are obtained as a rotation of old state variables. In fact, we can write

$$\begin{pmatrix} \hat{x}_1 \\ \hat{x}_2 \end{pmatrix} = \underbrace{\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}}_{R(\theta) \text{ Rotation Matrix}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

In fact, if we are not convinced yet:

$$\begin{aligned} \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} &= \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} \cos(\phi) \\ \sin(\phi) \end{pmatrix} = \\ &= \begin{pmatrix} \cos(\theta) \cos(\phi) - \sin(\theta) \sin(\phi) \\ \sin(\theta) \cos(\phi) + \cos(\theta) \sin(\phi) \end{pmatrix} = \begin{pmatrix} \cos(\phi + \theta) \\ \sin(\phi + \theta) \end{pmatrix} = \begin{pmatrix} \hat{x}_1 \\ \hat{x}_2 \end{pmatrix} \end{aligned}$$

For example, the matrix form of the biquadratic oscillator can be written as

$$\begin{pmatrix} \hat{x}_1 \\ \hat{x}_2 \end{pmatrix} = \begin{pmatrix} 2 \cos(\theta) & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

We have seen now that the equation we used can be now represented by an expression of this form. Let us generalise the matrix data model

$$\begin{pmatrix} \hat{x}_1 \\ \hat{x}_2 \end{pmatrix} = \underbrace{\begin{pmatrix} a & b \\ c & d \end{pmatrix}}_{\underline{A}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

Constrains on  $a, b, c, d$  to implement an oscillator are given by the Barkhausen criteria:

- $\det(\underline{A}) = 1 \Leftrightarrow ad - bc = 1$
- $|a + d| < 2$

The first condition implies unitary loop gain (the output neither decreases nor explodes); the latter (along with the former) implies that the eigenvalues of  $\underline{A}$  must be complex (actually, they must be complex conjugate, with unitary modulus). Another property that we want to point out is the  $b$  and  $c$  cannot be zero!

**Reminder: Barkhausen criteria**

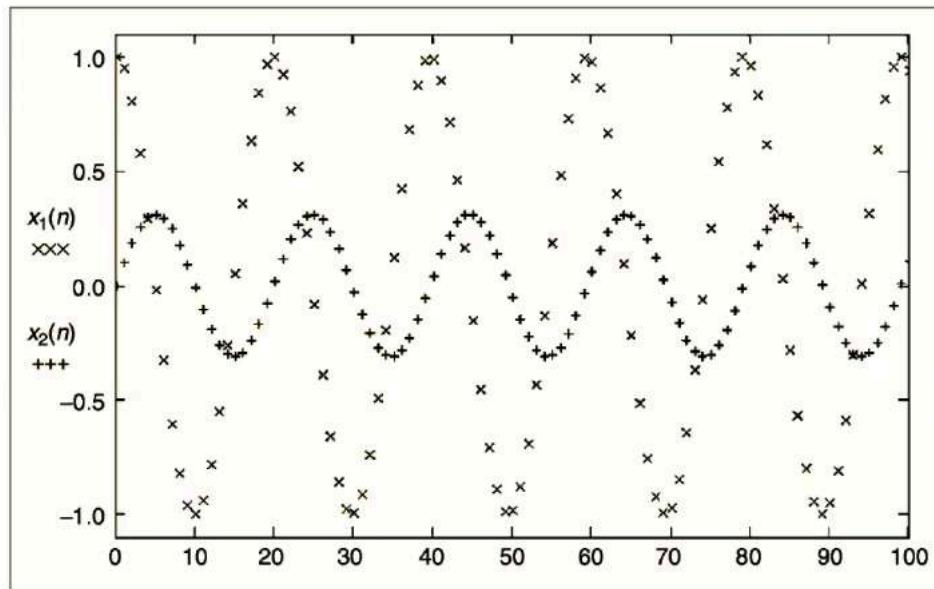
In electronics, the Barkhausen stability criterion is a mathematical condition to determine when a linear electronic circuit will oscillate. It was put forth in 1921 by German physicist Heinrich Barkhausen (1881–1956). It is widely used in the design of electronic oscillators, and also in the design of general negative feedback circuits such as op amps, to prevent them from oscillating.

Intuitively, if we think at the determinant, it represents the scale of the linear transformation that we are applying to our vector. Having the determinant equal to one means that this scale does not change. The version of the vector that we obtain after the rotation is similar in magnitude to the previous one.

Let us see an example of a generic form where matrix A is replaced with specific coefficients.

$$\begin{pmatrix} \hat{x}_1 \\ \hat{x}_2 \end{pmatrix} = \begin{pmatrix} 0.95 & -1 \\ 0.0975 & 0.95 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \quad \text{initial value: } \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

This means that if we implement a recursive time system described by the rotation matrix written above and we feed it with the initial condition  $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$  we get the following sinusoids.



So, the system (denoting by  $\underline{x}$  the initial state vector and by  $\underline{y}(n)$  the  $n$ -th output of the oscillator) can be represented by

$$\underline{y}(n) = \underline{\underline{A}}^n \underline{x}$$

A property that we can exploit to facilitate the matrix exponentiation is the so-called eigenvalue decomposition.

$$\begin{aligned} \underline{\underline{A}}^n &\stackrel{\substack{\text{Eigendecomposition} \\ \underline{\underline{A}} = \underline{\underline{Q}} \underline{\underline{D}} \underline{\underline{Q}}^{-1}}}{=} \underbrace{\left( \underline{\underline{Q}} \underline{\underline{D}} \underline{\underline{Q}}^{-1} \right)}^n = \underbrace{\left( \underline{\underline{Q}} \underline{\underline{D}} \underline{\underline{Q}}^{-1} \right)}_{n \text{ elements}} \underbrace{\left( \underline{\underline{Q}} \underline{\underline{D}} \underline{\underline{Q}}^{-1} \right)}_{\dots} \underbrace{\left( \underline{\underline{Q}} \underline{\underline{D}} \underline{\underline{Q}}^{-1} \right)}_{\dots} = \\ &= \underbrace{\underline{\underline{Q}} \underline{\underline{D}} \underline{\underline{Q}}^{-1}}_I \underbrace{\underline{\underline{Q}} \underline{\underline{D}} \underline{\underline{Q}}^{-1}}_I \underbrace{\underline{\underline{Q}} \underline{\underline{D}} \underline{\underline{Q}}^{-1}}_I \underbrace{\underline{\underline{Q}} \underline{\underline{D}} \underline{\underline{Q}}^{-1}}_I \dots \underbrace{\underline{\underline{Q}} \underline{\underline{D}} \underline{\underline{Q}}^{-1}}_I = \underline{\underline{Q}} \underbrace{\underline{\underline{D}} \underline{\underline{D}} \underline{\underline{D}} \dots \underline{\underline{D}} \underline{\underline{Q}}^{-1}}_{\underline{\underline{D}}^n} = \underline{\underline{Q}} \underline{\underline{D}}^n \underline{\underline{Q}}^{-1} \end{aligned}$$

By plugging this result into the equation that we wrote before, we get that

$$\underline{y}(n) = \underline{\underline{A}}^n \underline{x} = \underline{\underline{Q}} \underline{\underline{D}}^n \underline{\underline{Q}}^{-1} \underline{x}$$

We remind that  $\underline{\underline{D}}$  is a diagonal matrix with eigenvalues on the diagonal

$$\underline{\underline{D}} = \begin{pmatrix} e^{j\theta} & 0 \\ 0 & e^{-j\theta} \end{pmatrix}$$

Since it is a diagonal matrix, its exponentiation will consist in exponentiating the terms on the diagonal

$$\underline{\underline{D}}^n = \begin{pmatrix} e^{jn\theta} & 0 \\ 0 & e^{-jn\theta} \end{pmatrix} \quad \text{where } \theta = \arccos\left(\frac{a+d}{2}\right)$$

This interpretation of the matrix  $\underline{\underline{D}}$  allows us to see how can we perform a rotation of our state variables where  $\theta$  is nothing but the step angle of oscillation per integration.

In the equation,  $\underline{\underline{Q}}$  is the matrix that performs the change of basis

$$\underline{\underline{Q}} = \begin{pmatrix} 1 & 1 \\ \psi e^{j\phi} & \psi e^{-j\phi} \end{pmatrix} \quad \text{with } \psi = \sqrt{-\frac{c}{b}} \text{ and } \phi = \angle\eta$$

And, in turns

$$\eta = \frac{d - a + j\sqrt{4 - (a + d)^2}}{2b}$$

$\underline{\underline{Q}}$  and  $\underline{\underline{Q}}^{-1}$  are the maps between the “internal” and the “external” spaces.

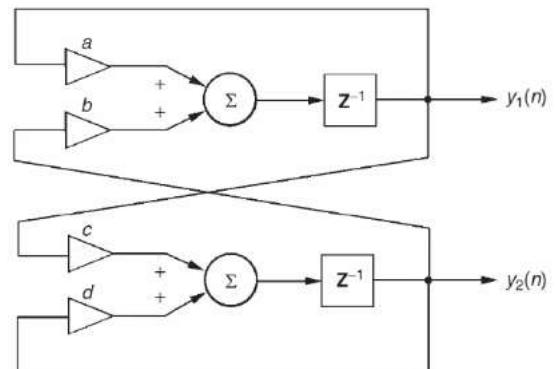
As any change of basis, the operation can be seen as made up by three main processes:

- Mapping onto the internal space
- Within the internal space: two complex conjugate internal rotations in opposite directions (to obtain a real oscillation)
- Mapping back onto the external space

Let us try now to interpret the rotation matrix:

- $\phi$  is the phase shift between the two state variables: when we want to obtain quadrature outputs, we impose  $\phi = \pm \frac{\pi}{2}$  that forces  $a = d$
- $\psi$  is the amplitude of the second state variable with respect to the first: when we want to obtain equal amplitudes, we impose  $b = -c$

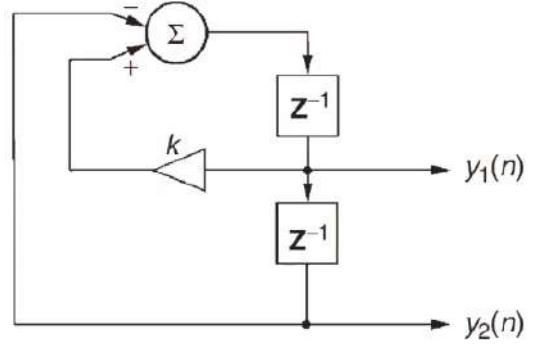
The DSP implementation of a generic oscillator is reported on the right.



### Biquadratic Oscillator

Quick overview on the corresponding implementation of the rotation matrix

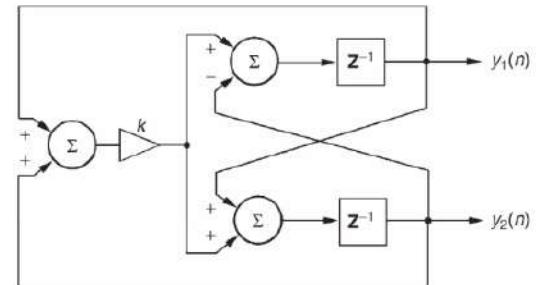
$$\underline{A} = \begin{pmatrix} k & -1 \\ 1 & 0 \end{pmatrix} \quad \text{with } k = 2 \cos(\theta)$$



### Digital waveguide Oscillator

Another structure can be

$$\underline{A} = \begin{pmatrix} k & k-1 \\ k+1 & k \end{pmatrix} \quad \text{with } k = \cos(\theta)$$



### Dynamic amplitude control

Algorithmic oscillators are “ballistic”, as they generate a signal in free evolution. This might be good for short period of time but not for long period of time since the Barkhausen condition is not respected anymore as some numerical error can stop the oscillation. In order to correct, or rather, to control those errors, an amplitude control would be needed. The standard approach to do it is the Automatic Gain Control (AGC). This method is basically based in the oscillator’s output power: when it is too much high the gain control intervenes.

$$P = \frac{x_1^2 - \frac{b}{c}x_2^2 - 2\sqrt{-\frac{b}{c}}x_1x_2 \cos(\phi)}{\sin^2(\phi)}$$

Porca miseria, it can be expensive... We come up with a simplified expression, valid for quadrature oscillators ( $\phi = \pm \frac{\pi}{2}$ )

$$P = \frac{x_1^2 - \frac{b}{c}x_2^2 - 2\sqrt{-\frac{b}{c}}x_1x_2 \cos(\phi)}{\sin^2(\phi)} \underset{\substack{\text{Quadrature} \\ \text{oscillators} \\ \phi=\pm\frac{\pi}{2}}}{=} \frac{x_1^2 - \frac{b}{c}x_2^2 - 0}{1} = x_1^2 - \frac{b}{c}x_2^2$$

Let us see how to perform the algorithm:

1. Measure the oscillator’s output power

$$P = x_1^2 - \frac{b}{c}x_2^2$$

2. Compute the gain factor

$$G = \frac{P_0^q}{P^q} \quad \text{where } P_0 = \text{set point power and } q = 0.5$$

This time again we approximate the results (there is Taylor somewhere)

$$G \approx 1 + q - q \frac{P}{P_0}$$

3. Scale the state variables by  $G$

This is how we cope with the amplitude modifications.

### Dynamic frequency control

In many applications we need to numerically control the frequency and magnitude trajectories of the oscillator. Changing an oscillator's frequency requires modifying the rotation matrix's  $k$  value for any new  $\theta$  frequency value. It must maintain amplitude control during dynamic frequency changes (magnitude depends on frequency). We need to update the coefficients in the power formula as the frequency changes (inefficient) and at the same time we need to control the magnitude trajectory of the oscillator... The problem disappears when we use a **coupled-standard quadrature oscillator**.

Oscillator	Mult./Iter.	Equi-Amplitude	Quadrature Output	$k =$	Rotation Matrix
Biquad	1	Yes	No	$2 \cos(\theta)$	$\begin{pmatrix} k & -1 \\ 1 & 0 \end{pmatrix}$
Digital Waveguide	1	No	Yes	$\cos(\theta)$	$\begin{pmatrix} k & k-1 \\ k+1 & k \end{pmatrix}$
Equi-amplitude-staggered-update	2	Yes	No	$2 \sin\left(\frac{\theta}{2}\right)$	$\begin{pmatrix} 1-k^2 & k \\ -k & 1 \end{pmatrix}$
Quadrature-staggered update	2	No	Yes	$\cos(\theta)$	$\begin{pmatrix} k & 1-k^2 \\ -1 & k \end{pmatrix}$
Coupled-standard quadrature	4	Yes	Yes	$\sin(\theta)$	$\begin{pmatrix} \sqrt{1-k^2} & k \\ -k & \sqrt{1-k^2} \end{pmatrix}$

### Wavetable sinusoidal oscillator

An alternative way to implement oscillator is the following: wavetables.

Pre-compute one period of a sinusoid and store the samples in a table (circular buffer) of length  $L$ , with a sampling period  $T_s$ , and signal period  $T_0 = LT_s$ . However, not all the sinusoids we want to implement have the same frequency  $\frac{1}{T_s}$ . So, how can we change the frequency?

The stolid solution could be to store the same waveform in tables of different lengths. A smarter solution could be storing a single wavetable of  $L$  (large) equidistant samples of the sinusoid and, for a desired frequency  $f_0$ , read the table each  $\Delta$  samples

$$\Delta = \frac{f_0 L}{F_s}$$

In order to implement fractionary steps, we need to priorly perform interpolation between samples.

## Time segment-based models

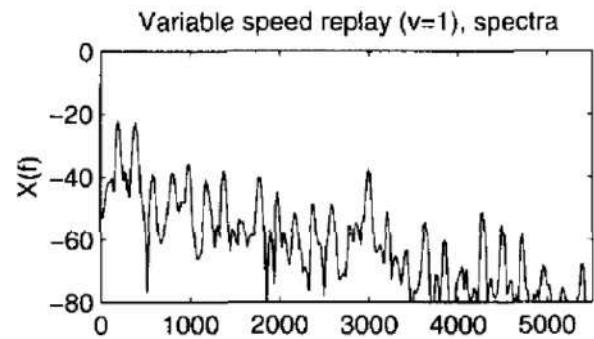
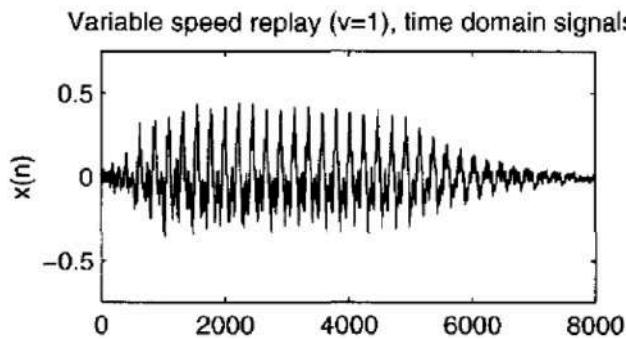
### Time compression/expansion

Time compression and expansion can be obtained by adopting a playback sampling frequency different from the recording one.

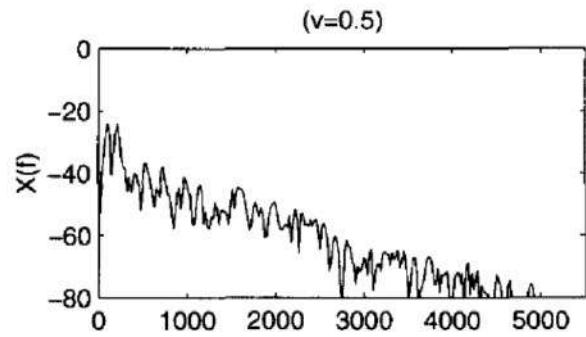
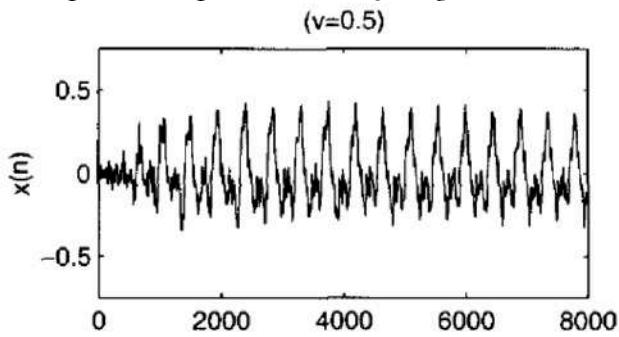
$$f_{s,play} = \nu f_{s,sec}$$

- if  $\nu < 1$ , the playback signal is an oversampled (by a factor  $\frac{1}{\nu}$ ) version of the recorded signal. We highlight that the sound quality depends on the interpolation filter adopted.
- if  $\nu > 1$ , the playback signal is a decimated (by a factor  $\nu$ ) version of the recorded signal. In order to avoid aliasing, before decimation the bandwidth of the recorded signals has to be limited to  $\frac{f_{s,play}}{2}$  by means of an anti-aliasing filter, that is nothing but a low pass filter.

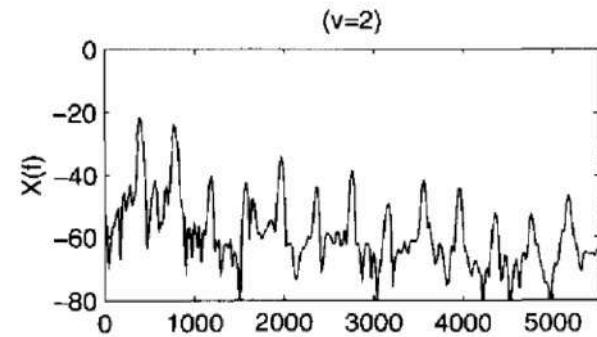
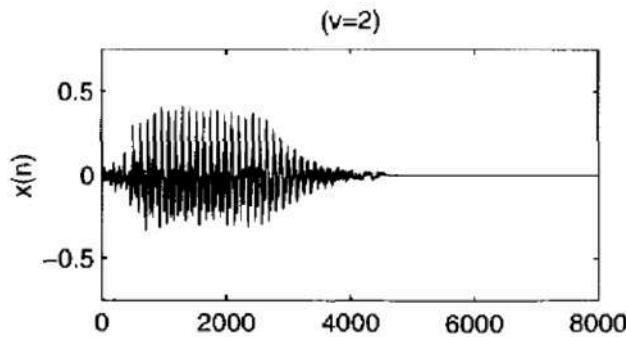
Let us see now how the spectrum and the waveform changes. Here we have the original waveform:



The expanded (reproduced at  $0.5x$ ) signal result as



And the compressed signal (reproduced at  $2.0x$ ) result as

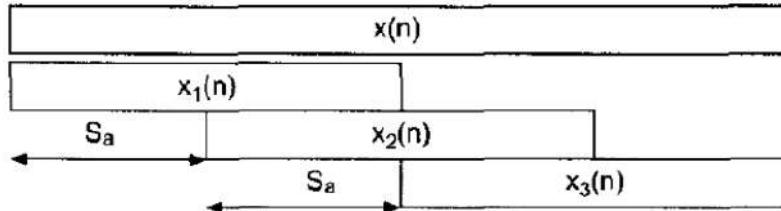


We notice is that we modify both the soundwave and the spectrum.

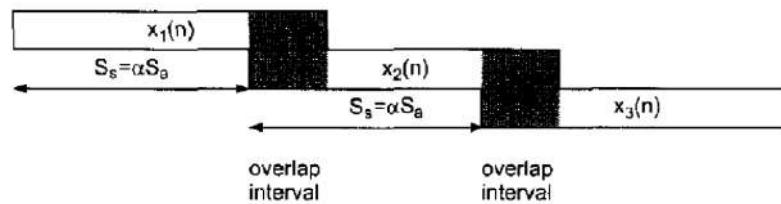
Is this desirable? Not at all, we would like to change the speed without touching the spectrum. To solve this issue, we come up with the SOLA and PSOLA techniques.

Let us briefly review how to perform the SOLA:

1. An input signal is segmented into block of length  $N$ , shifted by  $S_a$  samples



2. The blocks are repositioned with shift  $S_s = \alpha S_a$  where  $\alpha$  is the scale factor



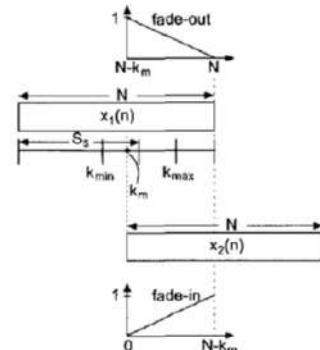
3. The computation of the cross-correlation between the overlapping segments  $x_{L,1}(m)$  and  $x_{L,2}(n)$  (of length  $L$ ) is performed

$$r(l) = \frac{1}{L} \sum_{n=0}^{L-l-1} x_{L,1}(n)x_{L,2}(n+l) \quad 0 \leq l \leq L$$

4. Extraction of the lag  $k_m$  at which  $r(l)$  has its maximum value

5. At  $k_m$  the signal  $x_1(n)$  is faded out while the signal  $x_2(n)$  is faded in.

6. A sum sample by sample is performed.

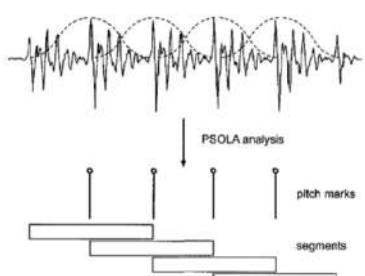


If we want to obtain a cleaner result, we can use the PSOLA technique: since the technique is based on the periodicities that are present in the signal, the segments can be perfectly synchronised in time avoiding discontinuities. One problem arises: we need to be aware of the position of these pitch pulses. If the signal is unpitched (white noise or similar) we cannot resort to the traditional SOLA.

In general, the PSOLA process is performed in two stages:

### *Analysis*

1. Determine the pitch period  $P(t)$  of the overall input signal
2. Determine pitch marks corresponding to glottal pulses
  - o at constant rate for unvoiced sound
  - o at pitch rate for voiced sound
3. Extract a segment centred at every pitch mark using Hamming window of length two pitch periods

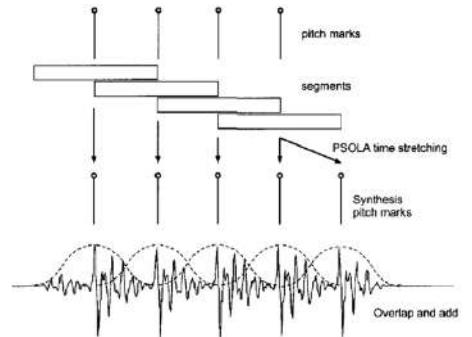


## Synthesis

In order to perform the synthesis, we follow these steps. For every synthesis pitch mark  $t_s$

1. Identify the corresponding analysis pitch mark  $t_a$  such that  
 $|at_a - t_s|$  is minimized (closest sample)
  - o the same analysis segment can correspond to different synthesis segments if  $\alpha > 1$
  - o some analysis segments may be discarded if  $\alpha < 1$
2. Overlap and add the selected segment
3. Determine next synthesis pitch mark in order to preserve the pitch

$$t_s^{next} = t_s + P(t_a)$$



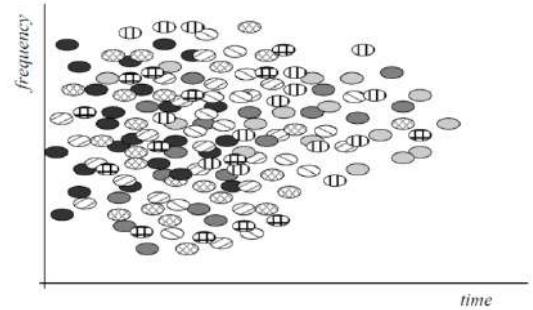
Problems:

It is not easy to identify the pitch and, in particular, is further difficult to do it in real time. Furthermore, when this technique is used for unvoiced signal, we can get unwanted periodicities if the sound comes from an unvoiced signal. To overcome this issue, we should detect the presence of transients to avoid repetition of such segments. In general, the SOLA approach is better for unvoiced sounds.

## Granular Synthesis

The basic idea behind granular synthesis is to create sound as a sequence of elementary (possibly overlapped) acoustic elements called **grains**. Originates from Gabor's idea of time-frequency spectrum. By the way, Gabor was the inventor of the STFT. The idea of granular synthesis could be seen as the composition of an STFT.

How can we formalise this concept with equations? Defining the signal model as a sum of weighted grains



Where a grain is defined as

$$g_{m,k}(t) = w(t - m\alpha T)e^{jk\beta\Omega t}$$

function that can be both real and complex

And, in turns:

- $w(t)$  is a synthesis window
- $m\alpha T$  is a time shift
- $k\beta\Omega$  is a frequency shift

How can we produce these grains? By performing the so-called sound granulation that consists simply in windowing signals. Obviously, the window  $w(n)$  is written in a way to provide smooth in and out transitions. The temporal length of the grain is critical, typically from 5 to 100 ms. Mathematically, a grain  $g_k(n)$  can be written as

$$g_k(n) = x(n + n_k)w(n)$$

Another possibility is to synthesise the grain from scratch

$$g_k(n) = w_k(n) \cos\left(\omega_k \frac{n}{f_s} + \phi_k\right)$$

Where:

- $w_k(n)$  is a Gaussian window of length  $L_k$  samples
- $\omega_k$  is the frequency of the grain

The sound is then synthetised by scattering grains on the time-frequency plane

$$s(n) = \sum_k a_k g_k(n - n_k)$$

In general, the synthesis can be characterised in two different ways according to how it is performed:

- *Synchronous granular synthesis*

In this type of synthesis, the density of grains in time remains constant as constant remains the time-step between successive grain generations.

- *Asynchronous granular synthesis*

The other possibility is to have grains that are irregularly distributed on the time-frequency plane.

This type of synthesis is used to simulate natural noisy sounds (statistical properties are more relevant than the exact waveform). For example, there are some pre-set representations that are called rain, rice and sand. With sand the pdf is higher than the rice one.

## Concatenative synthesis

A last technique used to synthetise sound (especially used to text-to-speech syntheses) is the concatenative synthesis. Here, the target sound is approximated by samples taken from a pre-recorded corpus of sounds. Sampled sounds are segmented into annotated units (attack, sustain, release portions). Units that match the target sound to be synthesised are selected based on high-level descriptors and then concatenated.

Nowadays this technique is not more used given the fact that text-to-speech tasks are absolved by AI.

## Spectral models: sinusoidal modelling of deterministic signals

### *Additive synthesis*

Additive synthesis consists simply in adding sinusoids at different frequencies. In general, the amplitude and the frequency are slowly varying over time. It is like composing an STFT.

We can synthetise signals by adding different components:

- Sinusoidal modelling of deterministic signals
- Synthesis-by-analysis approach to fit the parameters of a sinusoidal model to a target sound
- Inclusion of stochastic components and transients

In any case, we need to synthetise sinusoids: a fundamental block of our system will be a **discrete-time oscillator**.

For example, we can accurately reproduce the sound of a guitar generating the sinusoidal components and the filtering effect of the body.

Lucky, for a guitar,  $f_k$  and  $a_k$  are slowly varying over time so we can consider them constant in the window of time in which we are synthetising the sound. At the end, the signal can be synthetised as

$$s(n) = \sum_k a_k(n) \cos(\phi_k(n))$$

where

$$\phi_k(n) = 2\pi f_k(n)T_s + \phi_k(n-1)$$

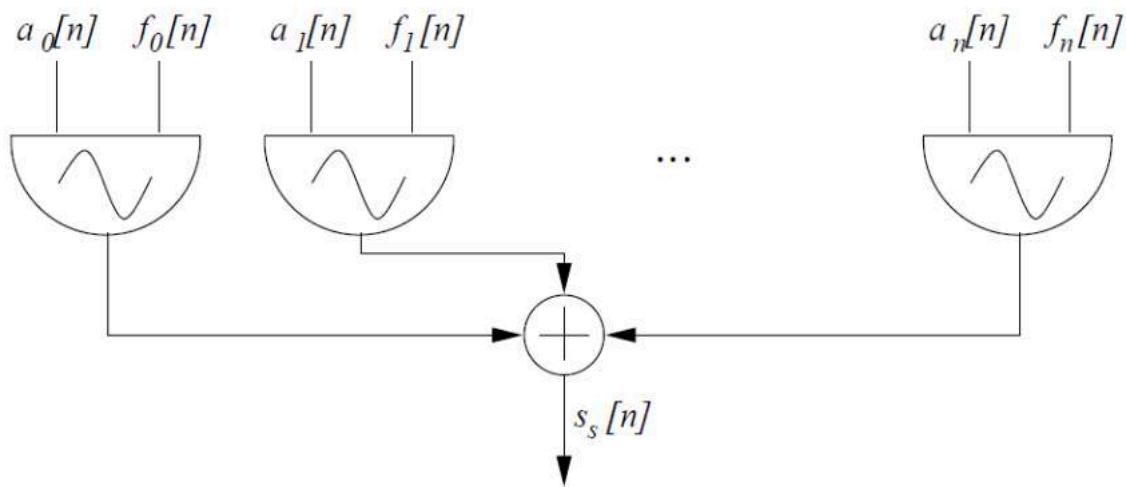
It is expressed in this way since the derivative of the phase is nothing but the frequency.

This approach can be used to synthesise many types of sounds both periodic (where  $f_k(n) \approx kf_0(n)$ , being  $f_0(n)$  the fundamental frequency) and aperiodic and enharmonic sounds (that can be represented only if their spectral energy is concentrated near discrete frequencies).

Pros	Cons
<ul style="list-style-type: none"> <li>The method is very intuitive: just a sum of sinusoids.</li> <li>It works, sometimes, also for non harmonic sounds: Marimba, xylophone and so on are not harmonic but can be described as a sum of sinusoids.</li> </ul>	<ul style="list-style-type: none"> <li>The model needs an extremely large number of control parameters: we need to know amplitude and frequency trajectories of all the partials of each note that has to be synthesised and, in addition, also their evolution in time since the amplitude trajectories generally depends on the strength (key velocity) of the note.</li> </ul>

### Implementation

The first kind of implementation of this technique is given by a set of oscillators.



The concept is immediate to understand but complex to implement: the oscillators are really large with lot of parameters that control it. How can we solve the problem making the system lighter? By working in the frequency domain.

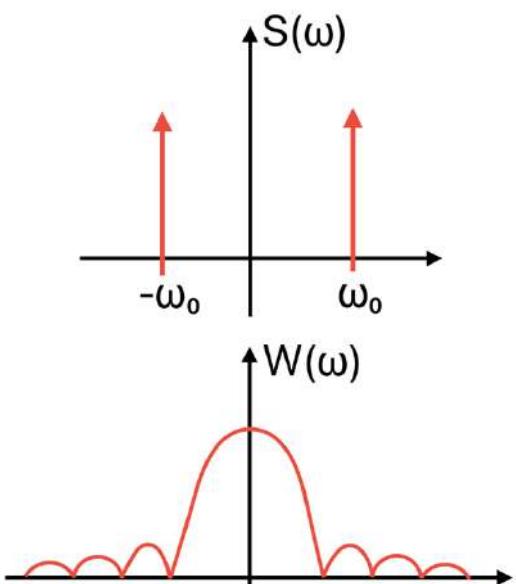
Considering a signal

$$s(n) = A \cos\left(\frac{\omega_0 n}{F_s} + \phi\right)$$

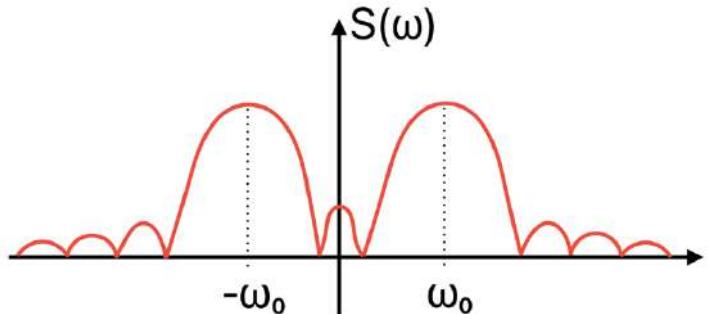
Its frequency response  $S(\omega)$  is described by two peaks located in  $\pm\omega_0$ . It is windowed by a function whose frequency response is reported on the right.

$$x(n) = w(n)s(n) = w(n)A \cos\left(\frac{\omega_0 n}{F_s} + \phi\right)$$

$$X(\omega) = \frac{A}{2} [e^{j\phi} W(\omega - \omega_0) + e^{-j\phi} W(\omega + \omega_0)]$$



We remind that the attenuation of the sidelobes with respect to the main lobe depends on the particular kind of window we are using. The product between the window and the signal returns a function like the one on the right. Not have a leakage of the energy beyond the frequency. How can we assure that the sound lobes are sufficiently short? By controlling the window length: the longest will be the number of samples, the shorter will be in time domain. But, in this way, we bump against the hypothesis of slowly varying signals. The longer is the window, the less valid is the hypothesis.



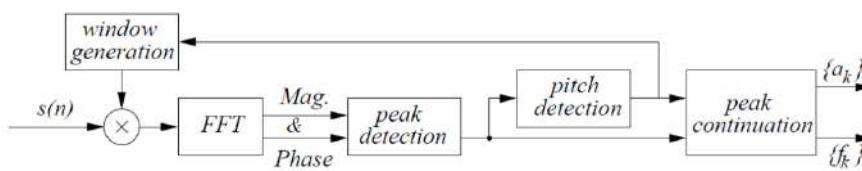
### Comparison between fft-based approach and time-domain approach

FFT-based approach is convenient when a high number of partials is desired (high frequency resolution). For this technique, the computational cost is determined by the inverse FFT, which cost does not depend on the number of components. We have to pay attention to find the right trade-off between temporal and frequency resolution: it is desirable to have short frames, so that frequencies and amplitudes can be assumed constant over a frame, hence there is no need for interpolation, but at the same time it is also desirable to have long frames, so that long FFT achieve higher frequency resolution and separation of sinusoidal components.

On the other hand, the time-domain approach is more flexible for instantaneous control of frequency and amplitude

### Synthesis-by-Analysis

As the name says, the synthesis by analysis is a process in which the synthesis is controlled by the signal itself, more precisely, the amplitude and the frequency envelops are extracted from short-time Fourier analysis (frame-by-frame). The process of analysis can be divided in three main steps:



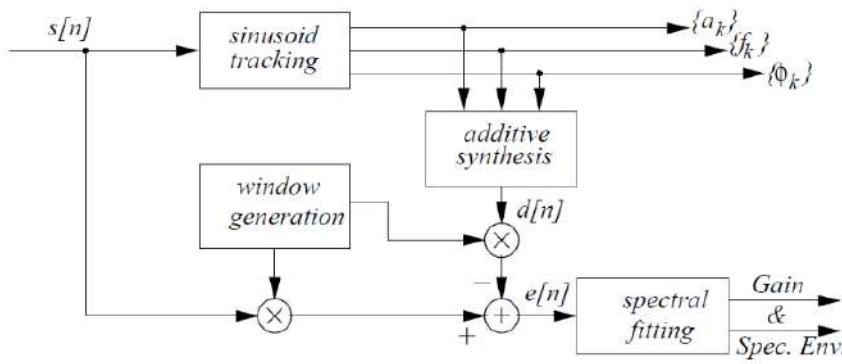
1. *Peak detection*: when the most prominent peaks are detected in the FFT.
2. *Pitch detection*: when the pitch is detected and the

window size is consequentially adapted to match it.

3. *Peak continuation*: in this phase, it is performed a building of trajectories to link peaks over time

At the end, the signal is synthetised by simply using additive synthesis.

### Additive model



As we said many times, it is quite easy to synthetise deterministic sounds as they are concentrated on a set of discrete frequencies. The same cannot be said for stochastic signals that are extremely broadband. How can we include them into the additive model? By simply summing the

stochastic component  $e(n)$  in time domain to the deterministic one  $s_s(n)$ . But, how can we determinate  $e(n)$ ? By performing the stochastic analysis: as we can see in the picture, it consists of performing a simple subtraction from which we get the signal  $e(n)$ .

$$e(n) = w(n)(s(n) - d(n))$$

Where  $w(n)$  is the signal representing the window and  $d(n)$  the re-synthesised deterministic component. We highlight that the window  $w(n)$  is needed to improve the trade-off between frequency resolution and time resolution: it is not mandatory that it has to be the same window used in the analysis. In fact, quite often it is shorter than the analysis one. What we get at the end, after a spectral filtering, are:

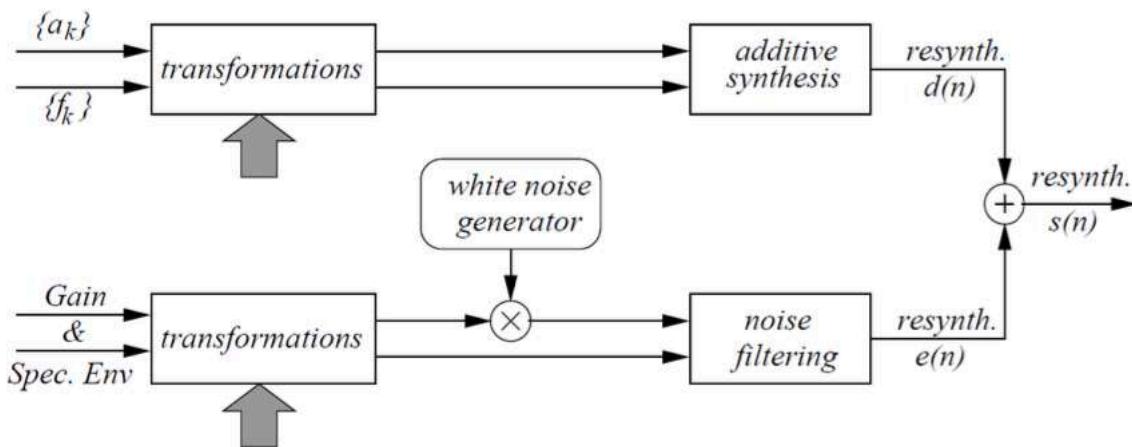
- The gain (energy with respect to the sinusoidal part) of the stochastic component
- Envelope of the spectrum

It is exactly what we need to design a filter  $H(k)$  that, multiplied by a white noise signal whose representation in the frequency domain is given by  $U(k)$ , returns exactly  $E(k)$  that is, in turn, the spectral representation of  $e(n)$ . Obviously, what we get it is a good approximation of  $e(n)$  not a representation of the “faithful” representation of  $e(n)$ , that is quite impossible to obtain due to the errors and approximation introduced in the process. We underline also that the filter has to change frame by frame following the evolution of the stochastic component of the actual signal.

Which is the algoritm that we can use to estimate the envelope? We see two of them:

- Performing a fitting in frequency domain
- LPC

Once we have extracted the envelope filter and estimated the gain of the stochastic comments, we can re-synthetise the signal. We can also mix and match different elements to obtain brand new signals.



Which is the advantage of having such a system at works? Once we analysed the original signal, we can reproduce it without the physical source that generated it. A system like that can allow us to:

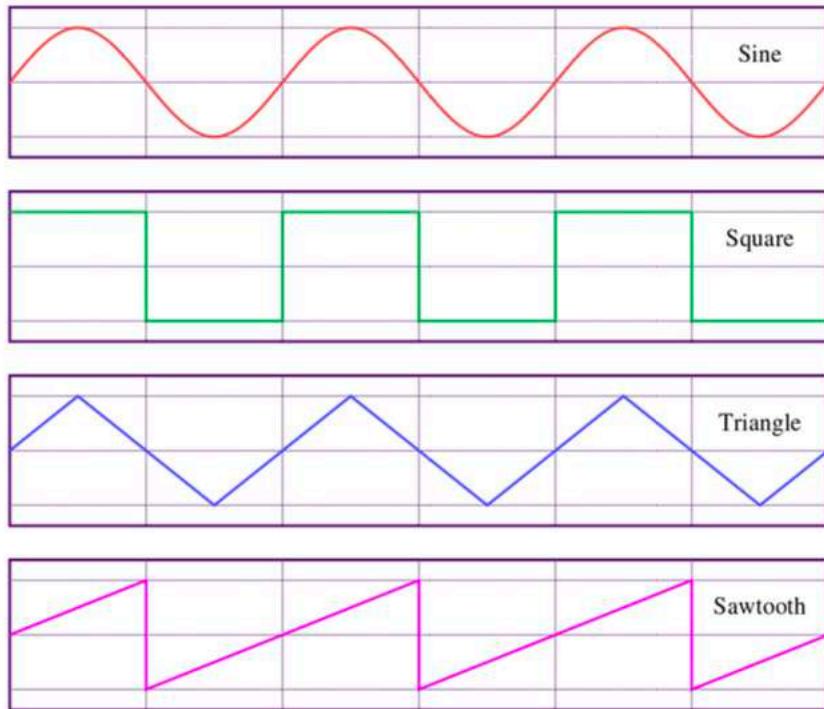
- Enhance the signal quality;
- Compress the signal: we have fewer parameter to transmit.

## References

- [1] Wikibooks. Sound Synthesis Theory / Oscillators and Wavetables, Apr. 2015. URL [http://en.wikibooks.org/wiki/Sound\\_Synthesis\\_Theory/Oscillators\\_and\\_Wavetables](http://en.wikibooks.org/wiki/Sound_Synthesis_Theory/Oscillators_and_Wavetables). Online; accessed 22/04/2015.
- [2] C. S. Turner. Recursive discrete-time sinusoidal oscillators. *IEEE Signal Processing Magazine*, 20(3):103–111, May 2003.
- [3] U. Zölzer, editor. DAFX Digital Audio Effects. John Wiley & Sons, 2002.
- [4] S. Roucos and A. M. Wilgus. High quality time-scale modification for speech. In Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), 1985.
- [5] C. Hamon, E. Mouline, and F. Charpentier. A diphone synthesis system based on time-domain prosodic modification of speech. In Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), 1989.
- [6] F. Avanzini and G. De Poli. Algorithms for sound and music computing, 2008.

## Wave Syntheses

In general, these are the three signals we can be interested in synthetise.



Let us try to analyse them one by one.

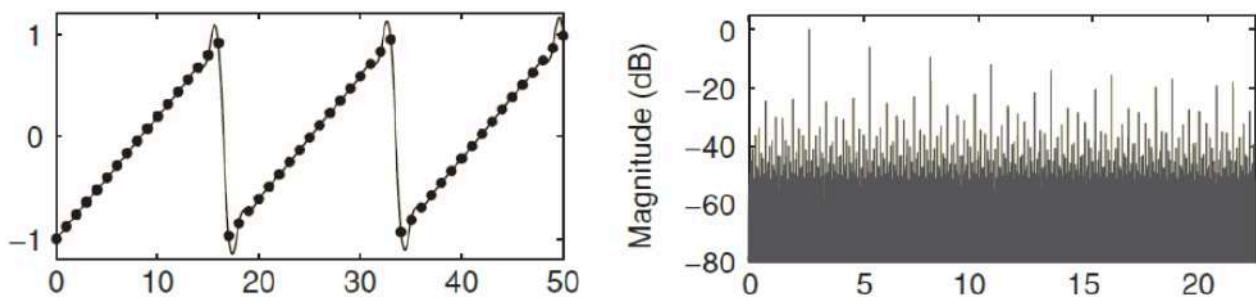
### Discrete-Time Sawtooth Oscillator

How can we synthetise this kind of signals for different frequencies? The most trivial solution is to count the samples, scale the sample index and dividing it by the sampling frequency.

$$s(n) = 2 \left[ n \frac{f_0}{F_s} \right]_{mod_1} - 1 \quad \text{for } n = 0, \dots, N$$

The overall counter increment per sampling interval is  $2 \frac{f_0}{F_s}$ , where  $f_0$  is the desired fundamental frequency and  $F_s$  is the sampling rate, and whenever the counter value is about to exceed one, 2 is subtracted from it to restart the ramp. Subtracting 1 from the whole waveform removes the DC offset.

As we said, it is the most trivial solution, but this simplicity comes also with problems. As we see in the following image, the sawtooth is not perfectly shaped, thing that becomes even more clear if we look at the spectral composition of the wave. There are no distinct peaks in correspondence of the multiples of the fundamental frequency but just a noisy representation.



This kind of spectrum is the typical depiction of the aliasing phenomena: all the spectral lines that are beyond the Nyquist limit (after the 8th partial if we consider a 44.1 kHz sampling frequency for the wave in the image) fold over and fill up the space between such spectral lines. We have to find some anti-aliasing strategies to avoid aliasing. The possible strategies are three:

1. Using bandlimited algorithms that produce a fixed number of harmonics (this includes additive synthesis);
2. Using quasi-bandlimited algorithms that sample a low-pass filtered continuous-time function (technique that has a moderate-high computational cost and memory usage);
3. Using algorithms based on spectral tilt modification (that modifies the spectrum to reshape aliasing into the non-audible range).

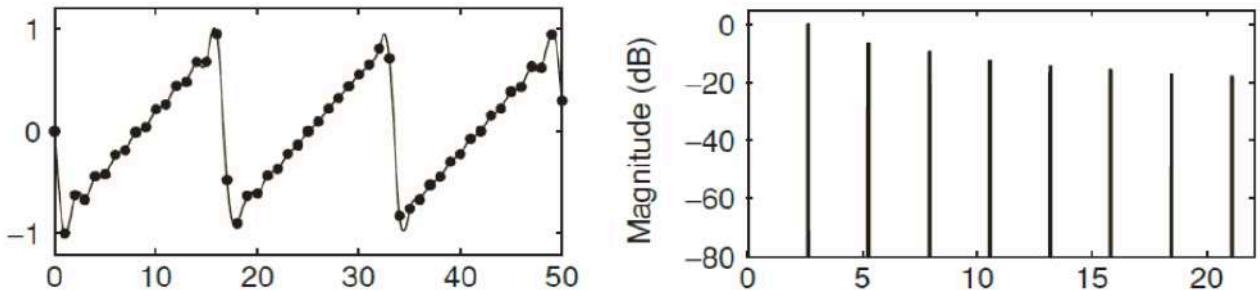
Basically, this algorithm applies an anti-aliasing filter.

Once the desired waveform is decomposed using the Fourier series, the same signal can be synthesised, using the first  $k$  harmonics (those that fall below Nyquist frequency), by means of the following formula:

$$s(n) = - \sum_{k=1}^K \frac{1}{k} \sin\left(2\pi \frac{f_0}{F_s} kn\right)$$

The minus in the expression allows us to obtain a ramp that goes from negative to positive values.

Unfortunately, this technique has two relevant issues: computational cost increases linearly with the number of partials  $K$  below Nyquist. This leads to an enormous computational cost if we want to synthesize a low-pitched sound (low  $f_0$ ) with all of its harmonics up to the Nyquist frequency. Furthermore, the spikes of the overall signal could be not periodic. This happens each time we do not have an integer number of periods inside the window we are considering as we can see below.



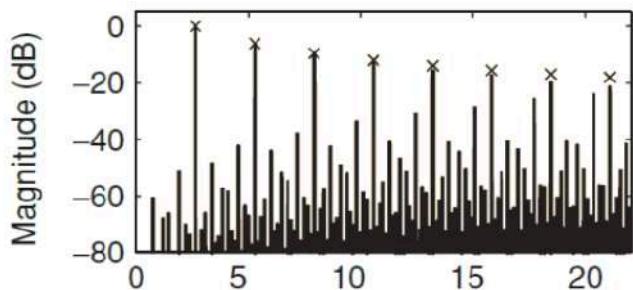
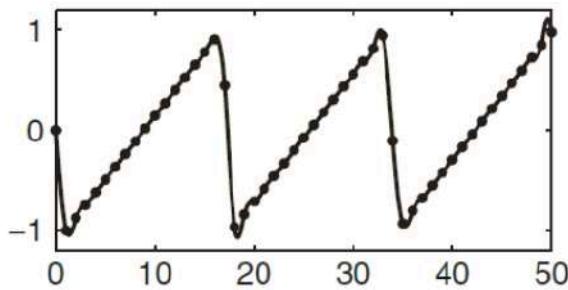
We can realise the sawtooth as a derivative of another signal: the derivative of a parabolic function. This technique, lack of fantasy, will be called Differentiated parabolic wave (DPW). What we do is synthesising a parabola and, by deriving it, obtaining a ramp.

Generation of the signal (parabolic):  $p(t) = \int_{-\frac{T_0}{2}}^t \tau d\tau \quad t \in \left[-\frac{T_0}{2}, \frac{T_0}{2}\right]$

Generation of the actual signal (sawtooth):  $s(t) = \frac{dp(t)}{dt}$

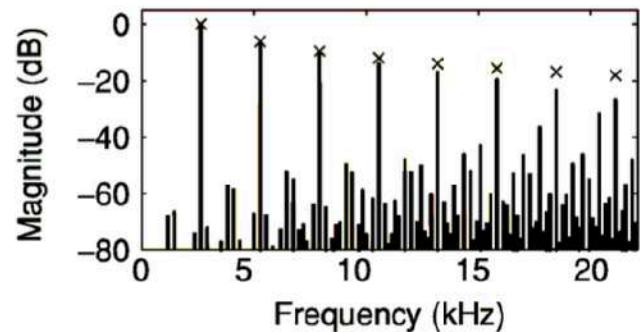
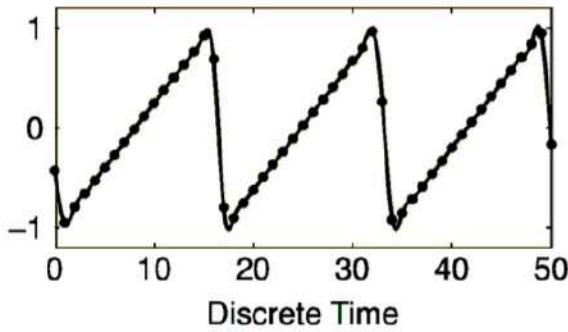
Why is this mechanism more suitable for us? Since the spectrum of  $p(t)$  decays about  $-12dB$  per octave: the frequencies that overcome the Nyquist's one will be reduced.

What we get is



If we compare this spectrum with the one we obtained before, we notice that the chaotic part is reduced with respect to the previous case.

The situation is even better if we use a 2x oversampled DPW method with a FIR decimation filter of order 6. What we get is

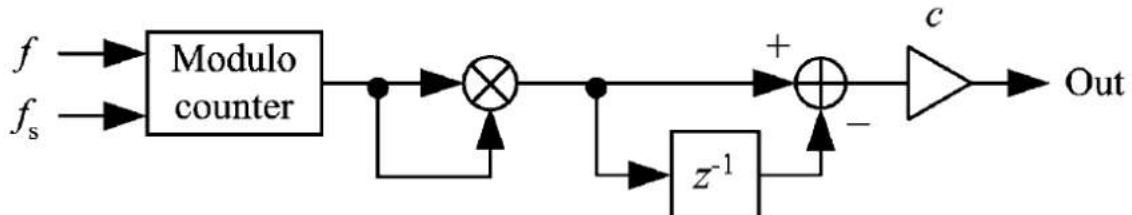


So, how can we implement it?

The simplest (and most efficient) version of the Differentiated Parabolic Waveform (DPW) algorithm generates the sawtooth waveform in four stages

1. The trivial sawtooth waveform is generated using a bipolar modulo counter;
2. The waveform is raised to the second power. We prefer not to do the integral since it is a bit demanding in terms of computation burden;
3. the signal is differentiated with a first difference filter with transfer function  $1 - z^{-1}$
4. the obtained waveform is scaled by factor  $c = \frac{f_s}{4f(1 - \frac{f}{f_s})}$

As the matter of fact, this is the scheme that we can use to implement the sawtooth generator



A first order FIR differentiator is described by  $D(z) = 1 - z^{-1}$

Last thing we have seen last time have seen last time is how to design a WPD technique to design a sawtooth wave.

### Square wave oscillator

The obvious way to design a square wave is simply to obtain it as the sign of a sine:

$$s(n) = \text{sign} \left( \sin \left( 2\pi \frac{f_0}{F_s} n \right) \right)$$

Where the fundamental frequency of the square wave will be equal to the frequency of the sinusoid. Let us see the spectrum by simply evaluating the Fourier Transform of the function

$$S(k) = \frac{4}{\pi} \sum_{k=1}^{\infty} \sin \frac{[(2k-1)2\pi \frac{f_0}{F_s} n]}{2k-1}$$

As we notice, the spectrum has lots of partials. Using this obvious algorithm, we bump into all the aliasing problem we saw for the sawtooth. So, how can we overcome this problem? By finding a function that, either integrated or derived, returns the square wave. We can see the square wave as a series of multiple constant: nothing but the derivative of a ramp.

### Triangle wave

The straight-forward way to obtain a triangle wave is to sample a continuous triangle wave  $tri(x) = 1 - |x|$ , for  $|x| < 1$ , 0 otherwise (non-periodic).

$$s(n) = \frac{2}{\pi} \arcsin \left[ \sin \left( 2\pi \frac{f_0}{F_s} n \right) \right] - 1$$

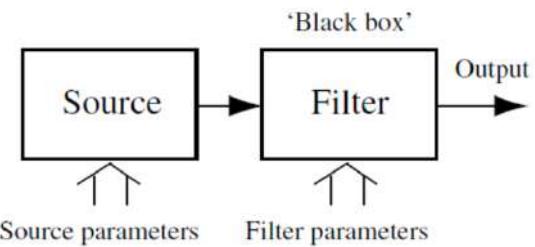
Whose spectrum is

$$S(k) = \frac{8}{\pi^2} \sum_{k=1}^{\infty} (-1)^{k-1} \frac{\sin \left[ (2k-1)2\pi \frac{f_0}{F_s} n \right]}{k^2}$$

This time again we bump into aliasing. In order to avoid the problem, we use some kind of WPD method. A comment we can do is that in the triangle wave we have no discontinuities: this means that the amplitude of the spectrum decays with  $\frac{1}{k^2}$  while for the square wave with  $\frac{1}{k}$ . Since it decays faster, the energy at the highest frequency will be smaller: this means that this kind of wave will get less aliasing problem than the square wave.

## Source-Filters Models

This technique dates back to analogue sound synthesis. It was used in speech synthesis and coding studies as well as in many music synthesizers. The structure is very simple: we have a source, an oscillator, and a filter, viewed as a black-box, that shape the signal coming from the source. Both of the blocks are controlled by time-varying parameters.

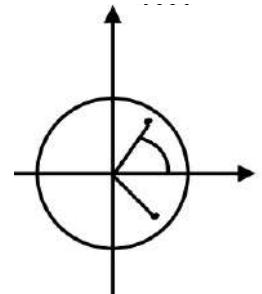


In general, to generate something interesting I need a source that is interesting since an LTI filter can just remove components from a signal and not add anything more. The source will be appropriated to the signal we want to generate. As a rule of thumb, that is not always true, we can say that the more complex the signal, the higher the number of harmonics in the source signal.

For what it concerns the filter, the most widely used in source-filter modelling are the resonant ones since they can select the harmonics we want to preserve from the source signal. More precisely, second-order IIR filter are used.

$$H(z) = \frac{b_0}{1 + a_1 z^{-1} + a_2 z^{-2}} \stackrel{\text{Decomposing the polynomial to find the two poles}}{=} \frac{b_0}{(1 - r e^{j\omega_c} z^{-1})(1 - r e^{-j\omega_c} z^{-1})}$$

Where  $r$  and  $\pm\omega_c$  are magnitude and phases of the poles and, in order to make the filter stable,  $r < 1$  must abide by the condition  $r < 1$ . A possible representation of the filter could be the one reported on the right.



If we want, we can derive the impulse response of our resonant filter. It comes in this close form equation

$$h(n) = \frac{b_0 r^n \sin(\omega_c(n+1))}{\sin(\omega_c)} u(n)$$

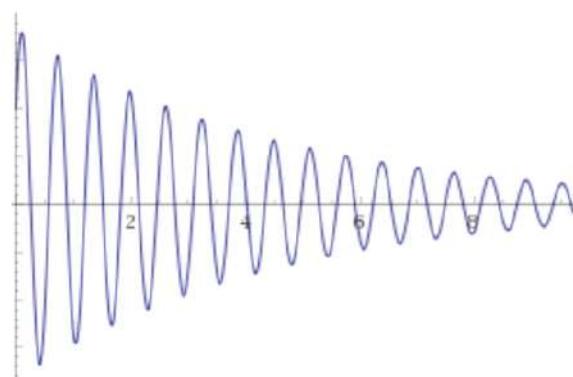
Where  $u(n)$  is the unit step and  $h(n)$  is a causal exponentially damped sinusoid.

We can think at the resonant filter as a very narrow band pass filter whose filter accuracy increases with  $r$ . Hence, it becomes sharper as  $r \rightarrow 1$ , its resonance is close to  $\omega_0$  and its parameters are

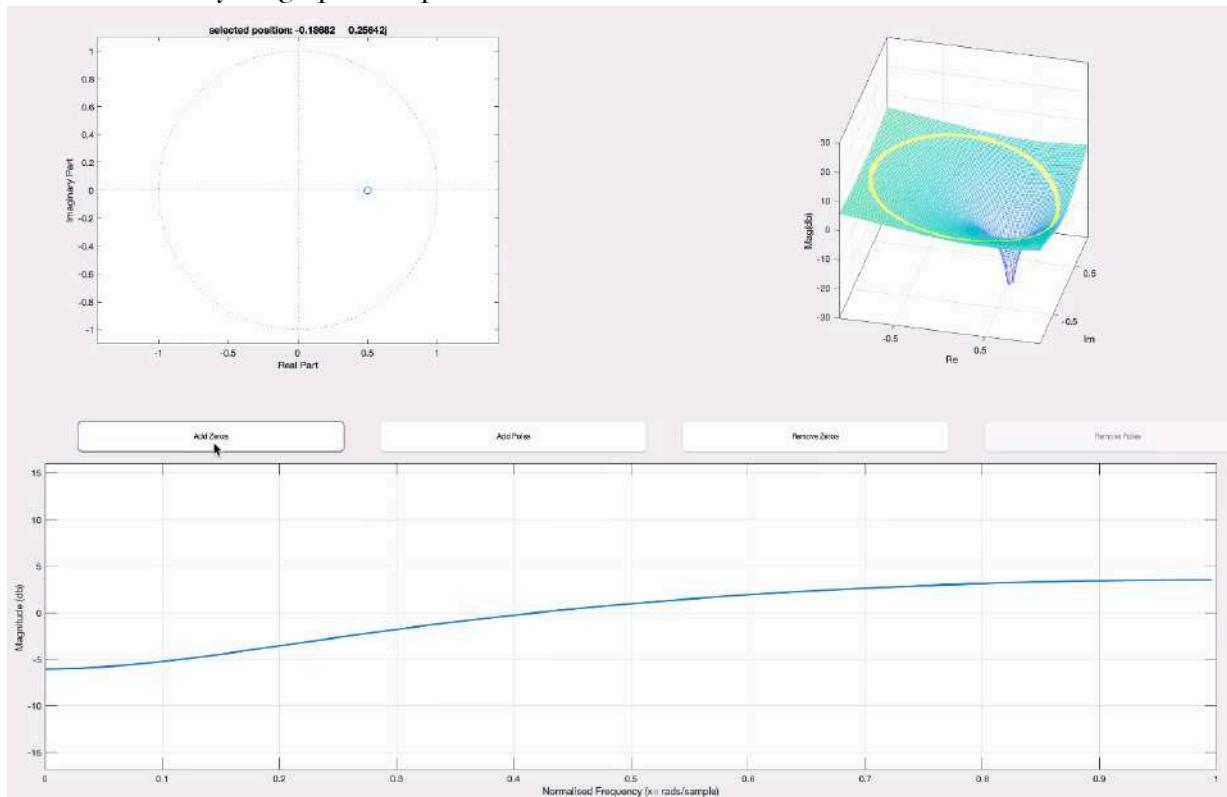
$$a_1 = -2r \cos(\omega_c) \quad | \quad a_2 = r^2 \quad | \quad b_0 = (1 - r^2) \sin^2(\omega_c)$$

where  $b_0$  is obtained imposing  $|H(e^{\pm j\omega_c})| = 1$

Which is the effect of this resonant filter? To produce a damped sinusoid whose frequency is related to the phase of the poles.

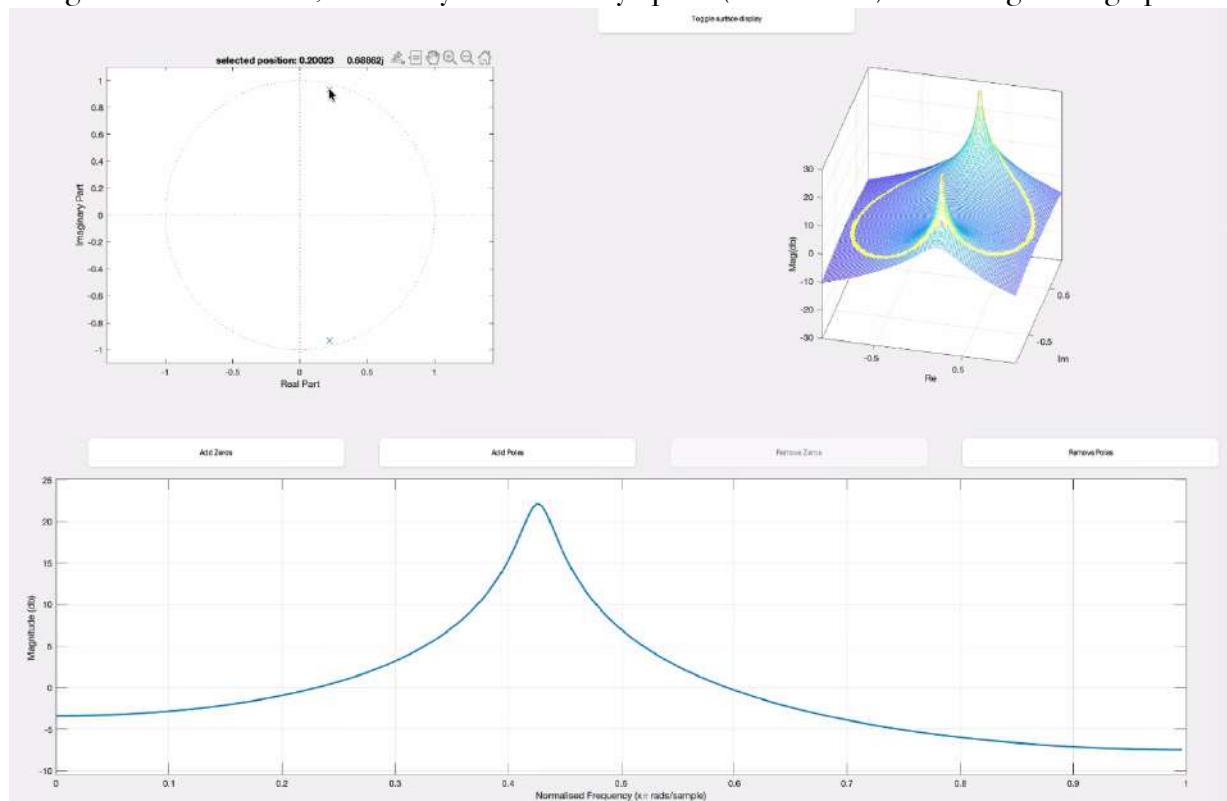


Reminder: intuitive representation of the magnitude of the Z-transform given the unit circle  
 Let us review briefly the graphical representation of the Z transform.



Adding a zero means to drop down to  $-\infty$  the whole (3D) transfer function. On the graph below (the magnitude) we can see the stretched representation of the unit circle. It is like if we cut the 3D graph along the unit circle, and we unwrap the cylinder on a plane.

This becomes evident, by placing a singularity (i.e. a zero) right on the outline of the 3d figure: the pick to  $\infty$  is right on the unit circle, that is why we have an asymptote (more or less) on the magnitude graph.



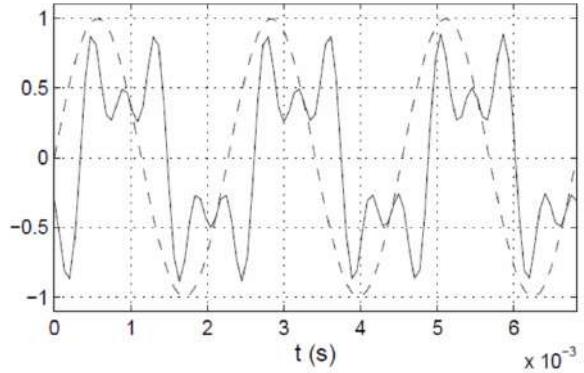
## Nonlinear modelling

To obtain something that is furthermore interesting, we can move to nonlinear modelling. As we know, a non-linear function applied to a signal can generate frequencies that are not present in the original signal. This allows us to enrich the spectrum. In order to measure these nonlinearities, we introduce the so-called harmonic distortion. Let us consider a signal  $x(n) = A \cos(\omega_0 n)$  that, been feed to a nonlinear system, has become

$$y(n) = \sum_{k=0}^N A_k \cos(k\omega_0 n)$$

The harmonic distortion is quantified through the total harmonic distortion

$$THD = \sqrt{\frac{\sum_{k=2}^N A_k^2}{\sum_{k=0}^N A_k^2}}$$



Theoretically, it is very interesting but practically not, since these systems are, in general, completely out of control: it is really hard to stabilise and control a non-linear model and, thus, predict the results. When this kind of models are used, the parameters are set empirically.

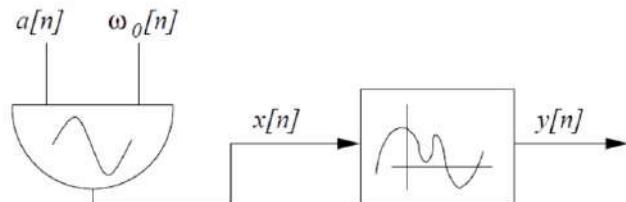
## Wave shaping

Let us consider for now a memoryless model in which the output depends only on the current input sample. A system like that can be modelled with the representation on the right.

It can be characterised by a *distortion function*

$F: \mathbb{R} \rightarrow \mathbb{R}$

$$y(n) = F(x(n))$$



In order to understand the effect of the harmonic distortion, we can consider the (truncated) Taylor expansion of the distortion function. We simply re-write the function as a series of polynomial that could approximate it.

$$y(n) = F(x(n)) = \sum_{i=0}^N a_i x^i(n)$$

This means that, if we consider for example the Taylor series up to the quadratic term  $a_2 x^2(n)$  we will have that:

- when  $x(n)$  is sinusoidal with frequency  $\omega_0$ ,  $x^2(n)$  has frequency component at  $2\omega_0$ .
- The square operation doubles the bandwidth of the input spectrum

## Overdrive and distortion as musical effects

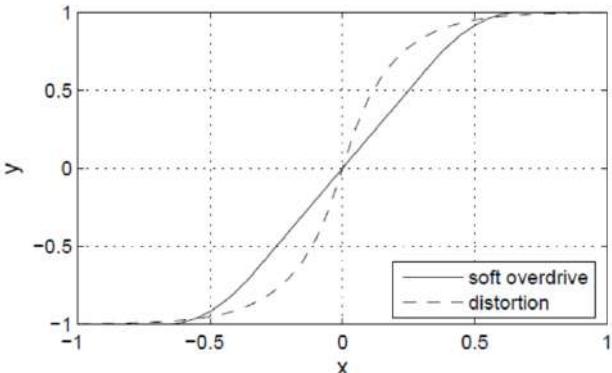
Analogical effects based on tubes or solid-state devices often are used to process guitar sounds in nonlinear fashion

- Overdrive is a musical term to designate nearly linear processing that can be driven into nonlinear regions when input dynamics is high. This is an almost linear effect.
- Distortion designates truly nonlinear processing. Decisively more non-linear.

These are maybe the two most famous non-linear effects. We consider now some nonlinear functions used to realise these effects.

Mathematically, they can be written with odd nonlinear function (so function that generate odd harmonics). These functions are approximately linear for low dynamics input values and, then, saturate with increasing input dynamics. We will have:

- Overdrive (softer distortion, in fact the function is almost linear)



$$F(x) = \begin{cases} 2x, & 0 \leq |x| \leq \frac{1}{3} \\ \text{sign}(x) \frac{3 - (2 - 3|x|)^2}{3}, & \frac{1}{3} < |x| \leq \frac{2}{3} \\ \text{sign}(x), & \frac{2}{3} < |x| \leq 1 \end{cases}$$

- Distortion (the slope is remarkable)

$$F(x) = \text{sign}(x) (1 - e^{-q|x|})$$

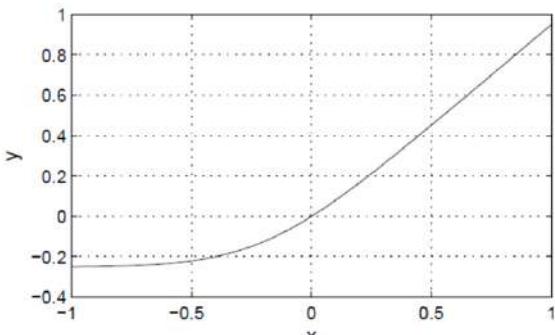
Where the coefficient  $q$  controls the amount of clipping

The more the system is linear, the more we go to a soft overdrive; the more the system is non-linear, the more we go to a distortion.

## Asymmetric distortion

Another kind of distortion we can take into account is the so-called asymmetric distortion. This kind of distortion resemble the effect of triode tubes. As the name suggest, positive and negative values are clipped in different ways (generate both even and odd harmonics)

$$F(x) = \frac{x - q}{1 - e^{-d(x-q)}} + \frac{q}{1 - e^{dq}}$$



Where:

- $q$  scales the range of linear behaviour
- $d$  controls the smoothness of transition to clipping

### Multiplicative synthesis

Another kind of interesting synthesis styles are the so-called multiplicative synthesis.

The simplest one is the ring modulation. Let us suppose having two signals  $x_1(n)$  and

$x_2(n)$ . If  $x_1(n)$  is the carrier ( $c(n) = x_1(n) = \cos(\omega_c n + \phi_c)$ ) and  $x_2(n)$  the modulating signal ( $m(n) = x_2(n)$ ), we get a signal showing a spectrum with two replicas of  $x_2(n)$  spectrum: one centred in  $+M\omega_c$  and another one in  $-M\omega_c$ . The two copies are called, respectively, Upper Side Band (USB) and Lower Side Band (LSB), reversed in frequency. We modulate the spectrum of the signal according to the frequency of the carrier.



$$S(\omega) = \frac{1}{2} [M(\omega - \omega_c)e^{j\phi_c} + M(\omega + \omega_c)e^{-j\phi_c}]$$

This phenomenon comes from telecommunication: if we want to broadcast more than one signal, we can assign different frequencies portion to different signals. In order to demodulate, we multiplicate the signal for the same carrier and then we low pass filter it in order to remove the unwanted frequencies.

With this kind of modulation, the problem comes when the signal has a larger bandwidth than the carrier frequency: in that case we have an overlapping between tails and, furthermore, a generative aliasing since a tail goes over the Nyquist limit. Mathematically speaking, when  $M(\omega)$  extends beyond  $\omega_c$  aliasing occurs and part of the LSB extends to the negative frequencies.

Example:

Given a sinusoidal carrier defined as  $c(n) = \cos(\omega_c n + \phi_c)$  and a modulated signal with fundamental frequency  $\omega_m$  and  $N$  harmonics,

$$m(n) = \sum_{k=1}^N b_k \cos(k\omega_m n + \phi_k)$$

multiplicative synthesis causes every spectral line  $k\omega_m$  to be replaced by two spectral lines at  $\omega_c - k\omega_m$  (in the LSB) and  $\omega_c + k\omega_m$  (in the USB)

$$s(n) = \sum_{k=1}^N \frac{b_k}{2} \{ \cos[(\omega_c + k\omega_m)n + \phi_k] - \cos[(\omega_c - k\omega_m)n + \phi_k] \}$$

If  $\omega_c - k\omega_m < 0$  for some  $k$ , the corresponding spectral line will be aliased around zero. The resulting spectrum has partial at  $|\omega_c \pm k\omega_m|$ ,  $k = 1, \dots, N$ .

When  $\frac{\omega_c}{\omega_m} = \frac{N_1}{N_2}$  is rational, with  $N_1, N_2 \in \mathbb{N}$  and mutually prime, the resulting sound is periodic, in fact all partials are multiple of a fundamental frequency equal to  $\omega_0 = \frac{\omega_c}{N_1} = \frac{\omega_m}{N_2}$ ,  $\omega_c$  will be the  $N_1$ th harmonic partial and  $\omega_m$  the  $N_2$ th harmonic partial.

When  $\frac{\omega_c}{\omega_m}$  is irrational, the resulting sound is enharmonic. If  $\frac{\omega_c}{\omega_m} = \frac{N_1}{N_2} + \epsilon$ ,  $\epsilon \ll 1$ , the spectrum is slightly enharmonic with fundamental frequency  $\omega_0 = \frac{\omega_m}{N_2}$ .

## Amplitude modulation

A slightly variation of the ring modulation is the so-called amplitude modulation. In this case, we introduce a parameter  $\alpha$  called amplitude modulation scale factor that controls the amplitude of sidebands.

$$s(n) = (1 + \alpha m(n)) c(n)$$

$$S(\omega) = C(\omega) + \frac{\alpha}{2} [M(\omega - \omega_c)e^{j\phi_c} + M(\omega + \omega_c)e^{-j\phi_c}]$$

In this case, the spectrum will contain also carrier spectral line.

## Synthesis by frequency modulation

Another way to synthetise signals can be obtained by nesting our modulating signal as a parameter of a sinusoid.

### Pros & Cons

#### Pros

- Versatile, it allows the generation of great variety of sounds.
- Limited number of control parameters.
- Low computational cost.

#### Cons

- Not suitable for analysis-by-synthesis scheme as parameters cannot be derived from analysis of real sounds.
- Parameters are not intuitive.

Let us see now how the frequency modulation works. It is given an FM oscillator

$$s(n) = a(n) \sin(\omega_c(n)n + \phi(n))$$

Where:

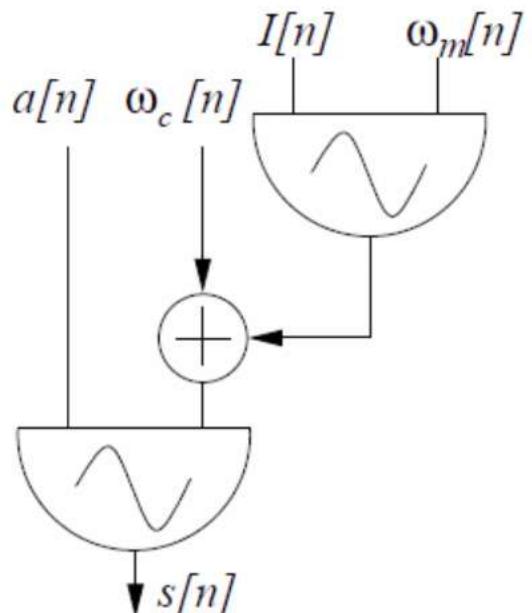
- $a(n)$  is the amplitude signal (control rate)
- $\omega_c(n)$  is the carrier frequency (control rate)
- $\phi(n)$  is the modulating signal (audio rate)

The iterative form of a FM module can be expressed by

$$\begin{aligned}\varphi(n) &= \varphi(n-1) + \omega_c(n)n + \phi(n) \\ y(n) &= a(n) \sin(\phi(n))\end{aligned}$$

Where  $\varphi(n)$  is the instantaneous phase of the FM module and  $\phi(n)$  is the sinusoidal modulating signal

$$\phi(n) = I(n) \sin(\omega_m(n)n)$$



The modulation process returns the following signal

$$s(n) = a(n) \sin[\omega_c(n)n + I(n) \sin(\omega_m(n)n)] = a(n) \sum_{k=-\infty}^{\infty} J_k(I(n)) \sin[(\omega_c(n) + k\omega_m(n))n]$$

Where:

- $J_k(I(n))$  is the  $k$ th order Bessel function of first kind
- Spectrum is composed by infinite number of partials at  $|\omega_c \pm k\omega_m|$
- Manipulation of  $I(n)$  produces an effect similar to low pass filtering.

What we conclude is that by modulating a sinusoid with a sinusoid, we obtain an infinite series of sinusoids that are sum together.

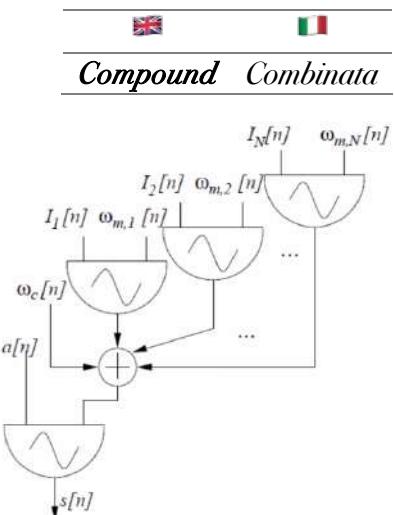
### Compound modulation

Another type of modulation is the compound modulation in which the modulating signal is composed by  $N$  sinusoids.

$$\phi(n) = \sum_{i=1}^N I_i(n) \sin(\omega_{m,i}(n)n)$$

That produces the signal

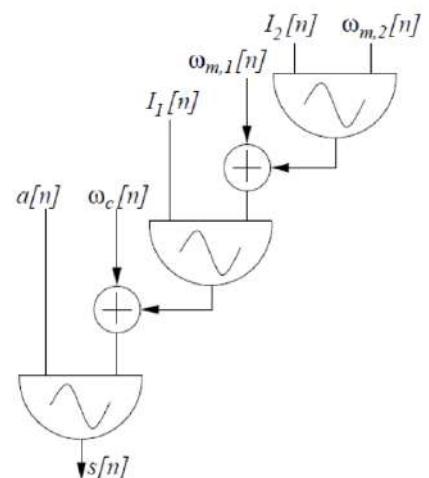
$$\begin{aligned} s(n) &= a(n) \sin \left[ \omega_c(n)n + \sum_{i=1}^N I_i(n) \sin(\omega_{m,i}(n)n) \right] = \\ &= a(n) \sum_{k_1, \dots, k_N}^{\infty} \prod_{i=1}^N J_{k_i}(I_i(n)) \sin \left[ \left( \omega_c(n) + \sum_{i=1}^N k_i \omega_{m,i}(n) \right) n \right] \end{aligned}$$



If the ratios between  $\omega_{m,i}$ 's are simple, the spectrum is slightly enharmonic. This configuration is used to simulate piano sound, using higher  $I_i(n)$  for lower tones (as low strings are enharmonic).

### Nested modulation

In the nested modulation, the sinusoidal modulator is itself modulated by a second one. It has a similar spectral structure to the compound with two modulators, but increased bandwidth.



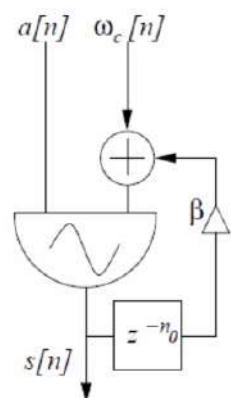
## Feedback modulation

Another thing that we can do is to insert loop in our scheme. In this case, we talk about feedback modulator.

Here we have  $\phi(n) = \beta s(n - 1)$  that produces the signal

$$s(n) = a(n) \sin(\omega_c(n)n + \beta s(n - 1))$$

If  $\beta$  increases,  $s(n)$  is periodic of fundamental frequency  $\omega_c$  and changes smoothly from a sinusoid to a sawtooth-like waveform.



## MATLAB

Just a couple of examples on Matlab

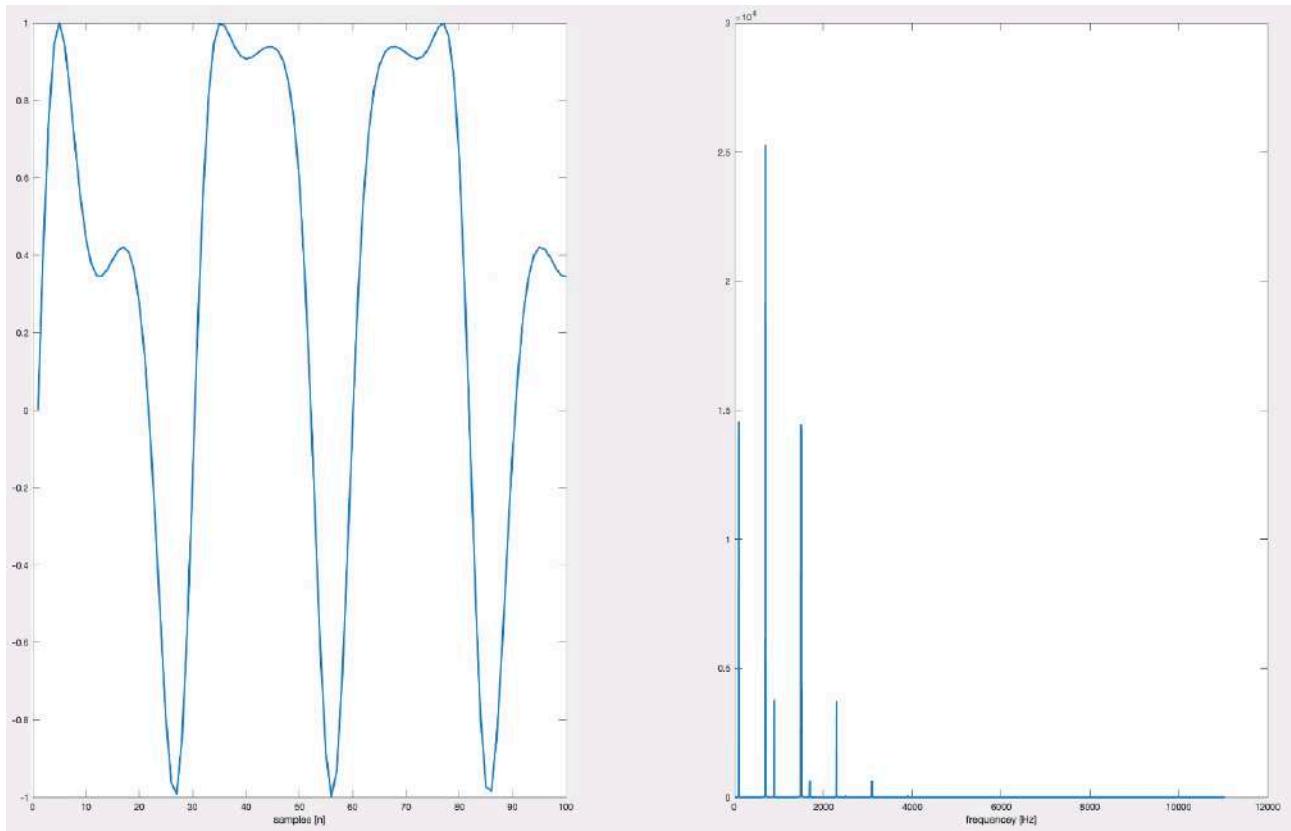
### Modulation on Matlab

```

1    clear all
2    close all
3    clc
4
5
6    Fs=22050;      % Sample Rate
7    fc=700;        % Carrier: 700 Hz
8    t=0:(1/Fs):3; % Duration
9
10   T = 3;         % 1 Simple Modulation - 2 Compound Modulation - 3 Nested Modulation
11
12  switch T
13    case 1
14      % Simple Modulation
15      I=1; % Modulation Index
16      fm=800;          % Modulation: 100 Hz
17      Fi = I*sin(2*pi*fm*t);
18
19    case 2
20      % Compound Modulation
21      I1 = 1;
22      I2 = .5;
23      fm1=100;          % Modulation: 100 Hz
24      fm2=810;          % Modulation: 2800 Hz
25      Fi = I1*sin(2*pi*fm1*t)+I2*sin(2*pi*fm2*t);
26
27    case 3
28      % Nested Modulation
29      I1 = 1;
30      I2 = 2;
31      fm1=100;          % Modulation: 100 Hz
32      fm2=800;          % Modulation: 2800 Hz
33      Fi = I1*sin(2*pi*fm1*t + I2*sin(2*pi*fm2*t));
34
35  end
36
37  s=sin(2*pi*fc*t+Fi);
38
39  Y = fft(s);
40  Y = abs(Y);
41
42  f = 0:Fs/length(Y): Fs - Fs/length(Y);
43
44  Y = Y(1:round(length(Y)/2));
45  f = f(1:round(length(f)/2));
46
47  figure(1)
48  subplot(121), plot(s(1:100)), xlabel('samples [n]')
49  subplot(122), plot(f,Y), xlabel('frequency [Hz]')
50  drawnow;
51  sound(s,Fs);
52

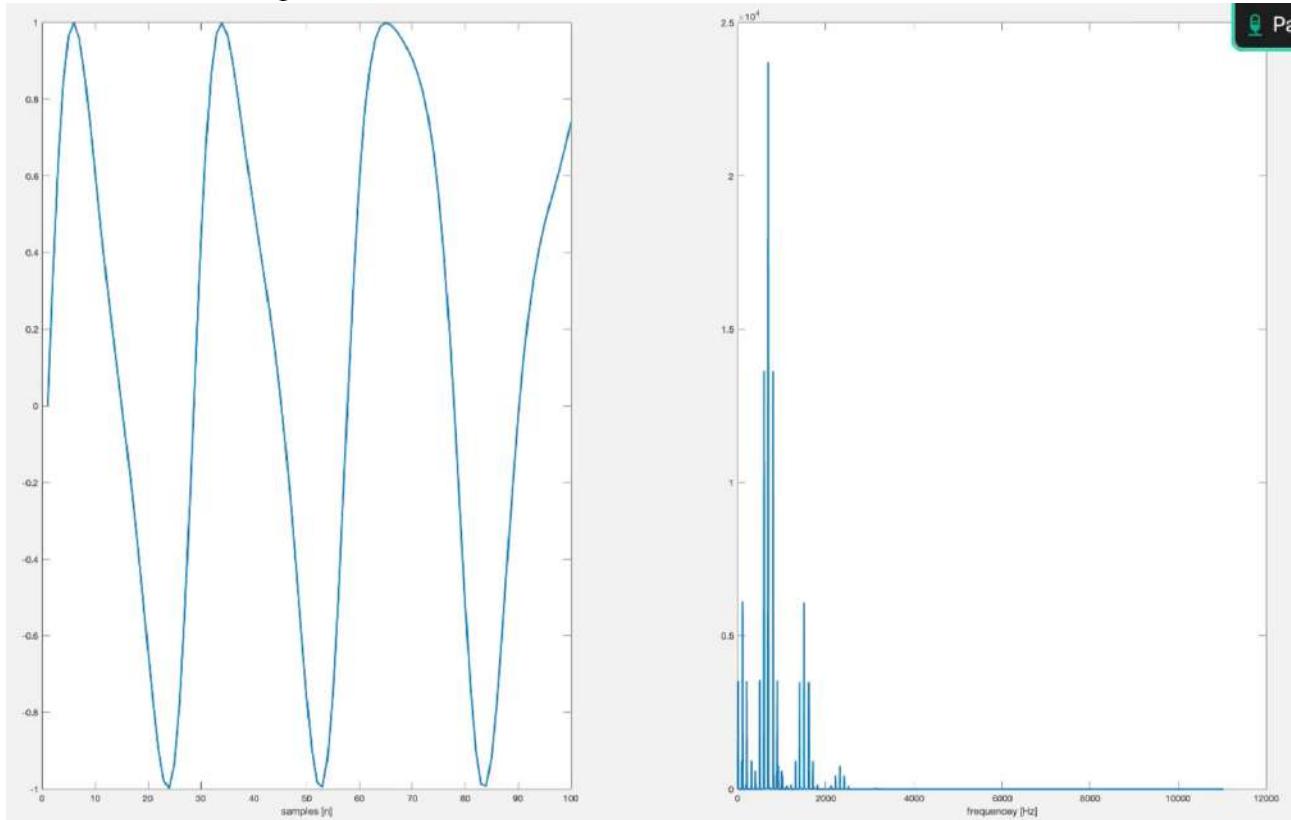
```

What we get is:



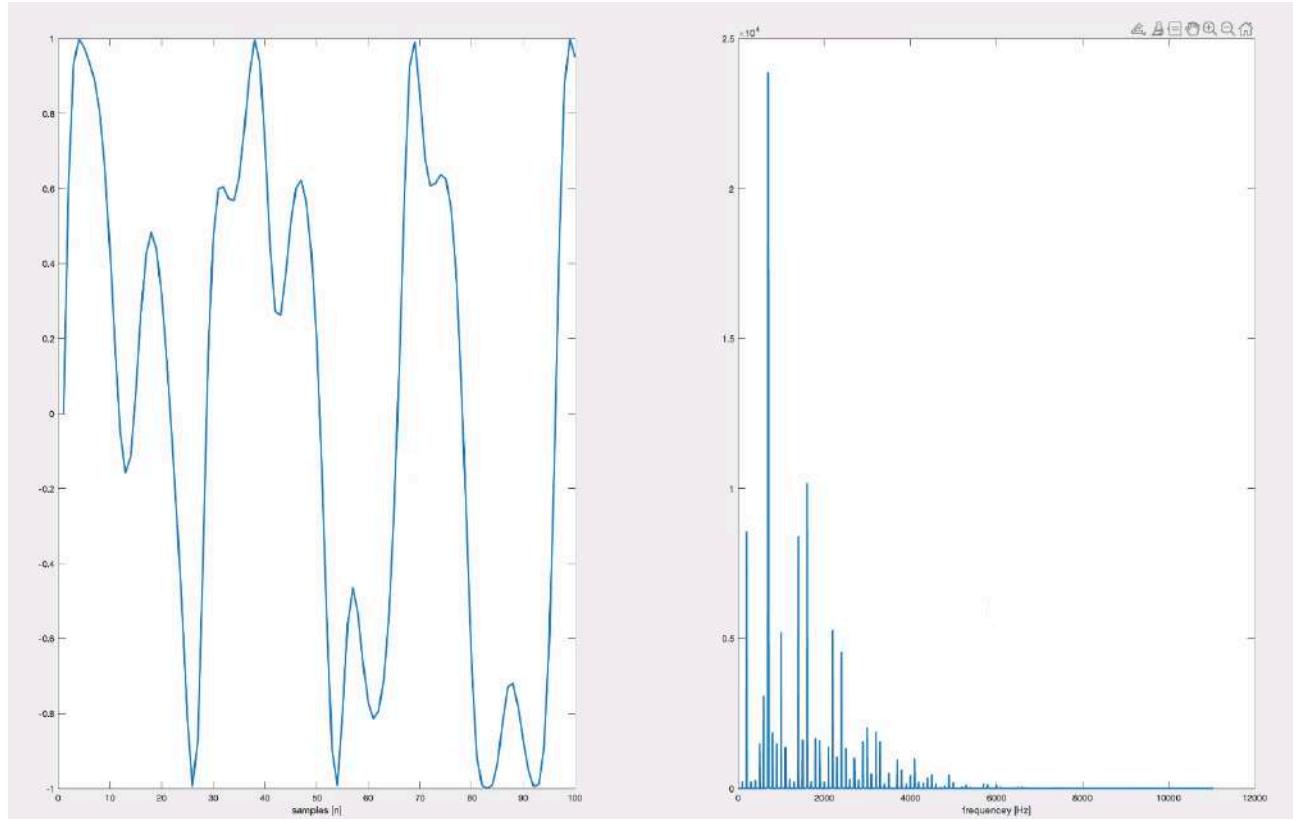
As we can notice, we generate more frequencies components. We can hear the roughness of the sound.

Case number two, compound modulation:



Much richer the spectrum.

Case number three, nesting modulation:



More relevant component of high frequencies.

## Linear predictive coding on Matlab

Let us see another interesting example:

This is an example of how to use linear predictive coding

```

1 % Example for cross synthesis of two sounds
2 clc, clear all, close all
3
4
5 % Load data
6 [exc, FS] = audioread('moore_guitar.wav'); % sound 1: excitation
7 env = audioread('Toms_diner.wav'); % sound 2: spectral e
8 % nv.
9
10
11 % Define Hanning windowing parameters
12 long      = 400;    % window length for calculation of coefficients
13 hopsize   = long/2; % hop size
14 w         = hanning(long); % window
15
16
17 % Define LPC parameters
18 order_env = 20; % order of the LPC
19 order_exc = 9;  % order for the excitation
20
21
22 % Trim audio samples to the same lenght and normalize amplitude
23 ly = min(length(exc), length(env));
24
25 N_frames = floor( (ly - long)/hopsize ) + 1;
26 ly = (N_frames-1)*hopsize + long;
27
28 exc = exc(1:ly);
29 exc = exc/max(abs(exc));
30 env = env(1:ly);
31 env = env/max(abs(env));

```

The order of excitation tells us for how long the system is excited.

```

32
33 %% Initialize output vector
34 out      = zeros(lv,1); % result sound (note: use of time difference equation
35 % which returns a vector of the same size of the input signal
36
37
38 %% Loop over each window and perform cross-synthesis
39 for j=1:N_frames
40
41     % Select analysis windows and perform auto correlation
42     k = hopsize*(j-1); % offset of the buffer
43     r_env = xcorr(env(k+1:k+long).*w);
44     r_exc = xcorr(exc(k+1:k+long).*w);
45
46     % Estimate LPC filter coefficients
47     [A_env, Dp_env] = levinson(r_env(long:end), order_env);
48     [A_exc, Dp_exc] = levinson(r_exc(long:end), order_exc);
49
50     % Compute LPC residuals
51     e_env = filter(A_env, 1, env(k+1:k+long));
52     e_exc = filter(A_exc, 1, exc(k+1:k+long));
53
54     % Synthesize output signal and write into buffer
55     out(k+1:k+long) = out(k+1:k+long) + w.*filter(1,A_env,e_exc*sqrt(Dp_env/Dp_exc));
56 end
57
58
59 %% Normalize output signal amplitude
60 out_norm = .99* out/max(abs(out)); % scale for wav output
61
62
63 %% Listen to the result
64 display('press a key to listen the excitation signal');
65 pause
66 soundsc(exc, FS);
67
68 display('press a key to listen the vocal (envelope) signal');
69 pause
70 soundsc(env, FS);
71
72 display('press a key to listen the cross-synthesis resulting signal');
73 pause
74 soundsc(out_norm, FS);
75

```

We can hear Suzanne Vega that sings with the voice of a guitar.

## References

1. Wikibooks. Sound Synthesis Theory / Oscillators and Wavetables, Apr. 2015. URL [http://en.wikibooks.org/wiki/Sound\\_Synthesis\\_Theory/Oscillators\\_and\\_Wavetables](http://en.wikibooks.org/wiki/Sound_Synthesis_Theory/Oscillators_and_Wavetables). Online; accessed 22/04/2015.
2. V. Valimaki and A. Huovilainen. Antialiasing oscillators in subtractive synthesis. IEEE Signal Processing Magazine, 24 (2):116–125, Mar. 2007.
3. V. Valimaki. Discrete-time synthesis of the sawtooth waveform with reduced aliasing. IEEE Signal Processing Letters, 12(3):214–217, Mar. 2005.
4. F. Avanzini and G. De Poli. Algorithms for sound and music computing, 2008.

## Source-Based Methods

### Origins

The proposal of using numerical simulations of the wave equation for sound synthesis made by Hiller and Ruiz in 1971 can be considered the game changer of sound synthesis. They declared:

*[...] This is a completely new approach to electronic sound synthesis insofar as the starting point is the physical description of the vibrating object [...]*

The numerical approach has been perfected by McIntyre, Schumacher and Woodhouse in 1983. Their improvement involved the usage of non-linear maps for modelling the generation of self-sustained oscillations in musical instruments. What they wrote in “On the oscillations of musical instruments” was:

*[...] a fast minicomputer could produce results at a cycle rate in the audible range. The result would perhaps have some novelty: an electronic musical instrument based on a mathematical model of an acoustic instrument [...]*

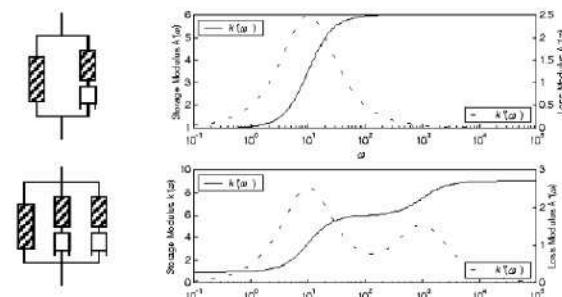
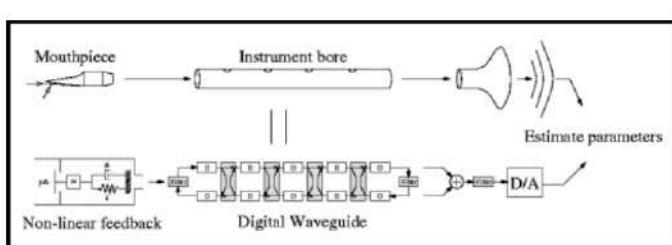
Today, we are able to perform real-time physical modelling on low-cost platforms.

### Physical Model Synthesis (PMS)

Source modelling focuses on the acoustic mechanism behind tone generation rather than the acoustic signal itself. Therefore, Timbral complexity is embedded in the model, not carried within the control signals (e.g. additive or granular synthesis). The algorithms tend to be computationally intensive; this is because there is an interaction between physical objects. This means that the control parameters (masses, hardness/softness characteristics, blowing pressure, lengths) are limited but highly informative and more difficult to process with respect to pure and plain mathematical expressions. Since we cannot use supercomputers to implement this kind of things, we need sophisticated modelling paradigms for efficient and flexible implementations (Finite Differences are for simulation). What we will need, will be:

- Modal Synthesis (for modelling and controlling resonances)
- Hybrid lumped/distributed modelling (e.g. WaveGuide Networks + Wave Digital Structures)

The fact that we model instrument instead of sounds comes with some advantages. Firstly, the parameters are not so much, and we are able to attribute to them a precise physical meaning or feeling (for example, we physically know the sensation of “hardness” thinking about the response of a string as we have experimented with our hands playing a guitar). This allows us to predict the result we will get counting on actual past experiences.



On the other hand, the things are not as easy as it could seem. Building physical models means understanding PMS is much more than mere simulation, understanding and implementing physical laws

(modelled in functional blocks that have to interact one with each other), overcoming algorithmic difficulties (discretisation, iterative solvers, ...), building different model for each type of interaction (this point can be avoided by learning through modular design as DWG, WDF), truly understanding how sounds are produced in nature, being aware that “the devil is in the detail” and it is not always advisable to account for all detail in a physical fashion.

The very few commercial physical model (PM) synthesisers and SW products only focus on presets and imitative synthesis. There is a great deal more beyond that:

- Lego-like approach:
  - o Building sounds from scratch rather than in a differential fashion
  - o Block-wise approach
  - o An interface that lets you build a procedural model in your mind
- Exploring timbral spaces
  - o Finding the «timbral region» within the parameter space is not easy in PMS
- Pseudo-physical sound synthesis
  - o stretching laws of physics
  - o making new ones up
  - o building «impossible» instruments or morphing seamlessly btw them in a seamless fashion
- Learning from physics how to design non-physical models
  - o Physics-inspirednon-physicalmodeling

Let us see some commercial products

### *Yamaha*

- VL1 (1994) – expensive duo-phonic keyboard
- VL1m, VL7 (1994) – tone module and less expensive keyboard, respectively
- VP1 (prototype) (1994)
- VL70m (1996) – less expensive tone module
- PLG-100VL, PLG-150VL (1999) – plug-in cards for various Yamaha keyboards, tone modules, and the SWG-1000 high- end PC sound card
- YMF-724, 740, 744, 754, and 764 sound chips for inexpensive DS-XG PC sound cards and motherboards
- 1S-YXG1ooplus-VL Soft Synthesizer for PCs with any sound card. Likewise equivalent to an MU50 plus VL70m
- S-YXG1ooplus-PolyVL SoftSynth for then-powerful PCs (e.g. 333+MHz Pentium III), capable of up to eight VL notes at once (all other Yamaha VL implementations except the original VL1 and VL1m were limited to one, and the VL1/m could do two), in addition to up to 64 notes of XG wavetable from the MU50-emulating portion of the soft synth. Only sold in Japan



## Korg

- Prophecy (1995)
- Z1, MOSS-TRI (1997)
- EXB-MOSS (2001)
- OASYS PCI (1999)
- OASYS (2005) with some modules, for instance the STR-1 plucked strings physical model
- Kronos (2011) same as OASYS



## Technics

- WSA1 (1995) PCM + resonator



Technics WSA1

## Seer Systems

- Creative WaveSynth (1996) for Creative Labs Sound Blaster AWE64.
- Reality (1997) - one of the earliest professional software synthesizer products by Dave Smith team

## Cakewalk

- Dimension Pro (2005) - software synthesizer for OS X and Windows XP

## Generalmusic

### Realpiano Pro (1995)

- Mix sampling and physical modelling for acoustic pianos

### Promega 2 (1998)

- As above, plus:
- Full physical modelling for electric pianos

### Promega 3 (2000)

- DRAKE (Dsp-Risc-Advanced-Keyboard-Engine), 32bit, 50Mips
- "Damper Physical Model"
- "Natural String Resonance": PM of sympathetic resonances
- "Advanced Release Technology": PM of dampers
- "FADE": modelling spectral changes associated to various key velocities

PM developed at CSC-UniPD



Generalmusic Real Piano Pro



Generalmusic Promega 2



Generalmusic Promega 3



Generalmusic Promega 2 (1997)



Generalmusic Promega 2+ (2017)

Finnish manufacturer Soundion



## A quick recap from physics

Let us now make a brief recap on physics reviewing the elementary 2<sup>nd</sup>-order physical oscillator. Simplest Lumped-parameter vibrating structure, used also for representing normal modes of oscillation of more complex systems. Ordinary differential equation (ODE) of the second-order (damped) oscillator

$$\ddot{x} + 2\alpha\dot{x} + \omega_0^2 x = 0$$

Whose general solution is

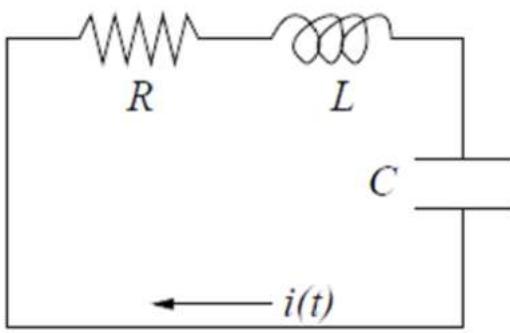
$$x(t) = a_0 e^{-\alpha t} \cos(\omega_r t + \phi_0) \quad \text{with} \quad \omega_r = \sqrt{\omega_0^2 - \alpha^2}$$

where  $a_0$  and  $\phi_0$  are uniquely determined by the initial condition  $x(0)$ ,  $x'(0)$

Let us report two examples of resonant systems in different domains:

Electrical Domain

RLC circuit



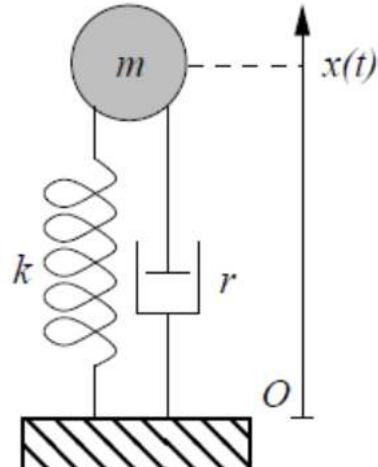
$$L \frac{d^2 i(t)}{dt^2} + R \frac{di}{dt} + \frac{1}{C} i = 0$$

The parameter of the solution will be

$$\alpha = \frac{R}{2L} \quad \omega_0^2 = \frac{1}{LC}$$

Mechanical domain

Mass, spring, damper system



$$m \ddot{x}(t) + r \dot{x}(t) + kx(t) = 0$$

The solution will be parametrised by

$$\alpha = \frac{r}{2m} \quad \omega_0^2 = \frac{k}{m}$$

Another example that is worth analysing more deeply, is the Helmholtz resonator. This is an example of a resonator system (second order oscillator) in acoustics. It is constituted by a body that presents a larger and a narrower section that produces a sound at a certain frequency. The overall structure is small with respect to the wavelength of the sound produced. Given this condition,  $p$  can be assumed constant.

Let us see now how the properties that leads to the resonance are divided between the different parts of the system:

- Resistive behaviour

Flow through opening caused by pressure difference. It depends on viscous and thermal losses, characteristics that are condensed into the fluid-dynamic resistance  $R$ .

$$\Delta p_{op}(t) = Ru(t)$$

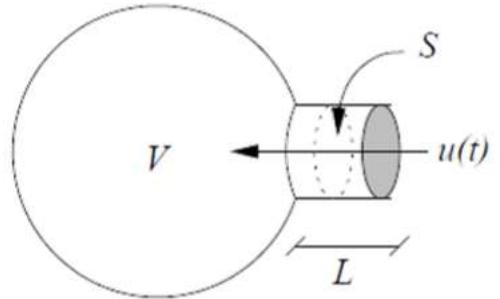
- Inertial behaviour (at the neck)

The space in the neck is so narrow that the air behaves “like” a solid block whose mass can be evaluated as

$$m = \rho_{air} S L$$

And, according to the Newton’s law, the pressure generated can be retrieved as:

$$\Delta p_{tube}(t) = \rho_{air} S L \dot{v}(t)$$



- Elastic behaviour (at the cavity)

Contraction of volume in the cavity caused by a pressure difference (variation of mass = difference in pressure)

$$-\rho_{air} c^2 \cdot \frac{dV}{V} = \Delta p_{cav}$$

The variation  $-dV$  of air volume must equal the time integral of the air flow

$$-dV(t) = \int_0^t u(t') dt' = \frac{V}{\rho_{air} c^2} \Delta p_{cav}(t) \Rightarrow S \Delta p_{cav}(t) = -\frac{\rho_{air} S^2 c^2}{V} \int_0^t u(t') dt'$$

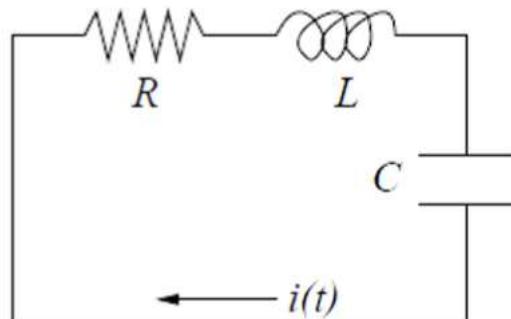
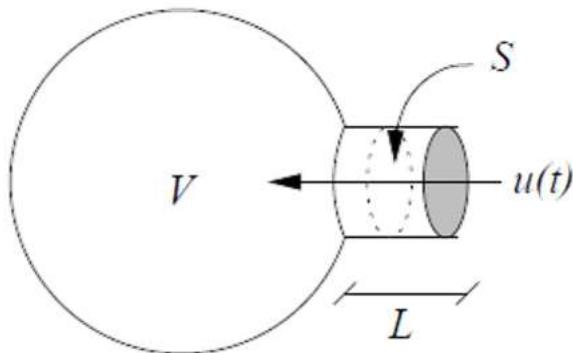
Modelled like a linear spring of stiffness

$$\frac{\rho_{air} S^2 c^2}{V}$$

Air mass and resistance at the opening involve the same flow  $u$ : they are connected in series. The flow  $u$  enters the cavity, so that the volume is in series with the other two. We can therefore represent it with an equivalent series circuit

$$(\rho_{air} S L) \cdot \ddot{x}(t) + R \dot{x}(t) + \frac{\rho_{air} S^2 c^2}{V} x(t) = 0$$

$$L \frac{d^2 i(t)}{dt^2} + R \frac{di(t)}{dt} + \frac{1}{C} i(t) = 0$$



## Kirchhoff variables

In physical systems it is always possible to identify pairs of variables (Kirchhoff variables) whose product is a power ( $W = K g \frac{m^2}{s^3}$ ). For example, In electrical systems we work with pairs of voltage-current (port) variables: ( $v, i$ ). Depending on the integro-differential relations basic elements can be defined

$$V(s) = R \cdot I(s) \quad | \quad V(s) = sL \cdot I(s) \quad | \quad V(s) = \frac{1}{sC} I(s)$$

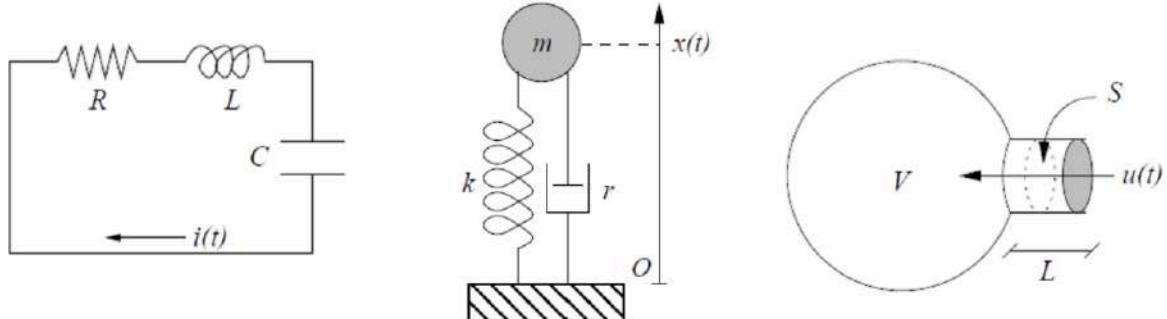
In general,

$$V(s) = Z(s)I(s)$$

Where  $Z(s)$  is called impedance and is generally defined as the ratio between the L-tf of voltage and current. We remind that the inverse of  $Z(s)$  is called admittance  $\Gamma(s)$ .

Let us summarise them

Electrical		Mechanical		Acoustic	
Current $i$ (A)		Velocity $v$ (m/s)		Flow $u$ (m <sup>3</sup> /s)	
Voltage $v$ (V)		Force $f$ (N)		Pressure $p$ (Pa)	
(Resistance) $R$		(Damping) $r$		(Opening) $R$	
(Capacitance) $1/sC$	$\left(\frac{\text{Kg}\cdot\text{m}^2}{\text{s}}\right)$	(Spring) $k/s$	$\left(\frac{\text{Kg}}{\text{s}}\right)$	(Cavity) $\rho_{\text{air}} c^2/Vs$	$\left(\frac{\text{Kg}}{\text{m}^4\cdot\text{s}}\right)$
(Inductance) $sL$		(Mass) $m \cdot s$		(Bore) $\rho_{\text{air}} LS s$	

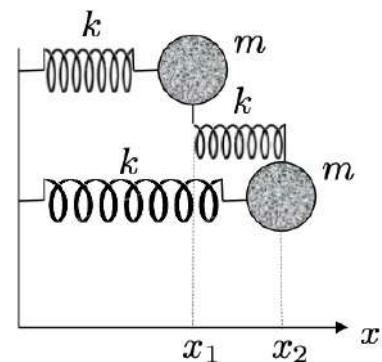


## Modal decomposition

More complex oscillating systems can be modelled as combinations of simple oscillators. In this context, the system can be reformulated using normal coordinates (normal modes of oscillation).

We report an example in which two point-masses (same mass  $m$ ) are connected to each other and to the “walls” through three springs (same stiffness  $k$ ). The model is the following:

$$\begin{aligned} m\ddot{x}_1(t) + kx_1(t) + k(x_1(t) - x_2(t)) &= 0 \\ m\ddot{x}_2(t) + kx_2(t) + k(x_2(t) - x_1(t)) &= 0 \end{aligned}$$



How can we decouple (make them independent one with the other) them?

By diagonalising the system.

$$\begin{aligned} q_1 &= x_1 + x_2 & \ddot{q}_1 &= -\omega_0^2 q_1 \\ q_2 &= x_1 - x_2 & \ddot{q}_2 &= -3\omega_0^2 q_2 \end{aligned} \quad \omega_0^2 = \frac{k}{m}$$

Normal modes  $q_i$  are uncoupled and the  $x_i$  are linear combination of such modes.

$$\begin{cases} m\ddot{x}_1(t) + kx_1(t) + k(x_1(t) - x_2(t)) = 0 \\ m\ddot{x}_2(t) + kx_2(t) + k(x_2(t) - x_1(t)) = 0 \end{cases} \rightarrow$$

$$\xrightarrow{\substack{\{q_1=x_1+x_2 \\ q_2=x_1-x_2}} \begin{cases} x_1 = \frac{q_1+q_2}{2} \\ x_2 = \frac{q_1-q_2}{2} \end{cases}} \begin{cases} \frac{m}{2}(\ddot{q}_1 + \ddot{q}_2) + \frac{k}{2}(q_1 + q_2) + kq_2 = 0 \\ \frac{m}{2}(\ddot{q}_1 - \ddot{q}_2) + \frac{k}{2}(q_1 - q_2) - kq_2 = 0 \end{cases} \xrightarrow{\omega_0^2 = \frac{k}{m}} \begin{cases} \ddot{q}_1 = -\omega_0^2 q_1 \\ \ddot{q}_2 = -3\omega_0^2 q_2 \end{cases}$$

This can be generalised to more complicated systems, made of  $N$  masses coupled through springs and dampers. We can generally rewrite the system in terms of normal modes of oscillation, and the oscillation of each point mass can be seen as a linear combination of  $N$  normal modes, each obeying the equation of a 2<sup>nd</sup>-order (damped) harmonic oscillator.

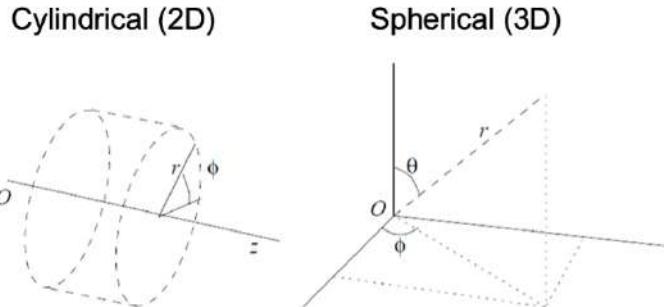
In general, a continuous vibrating system where a wave can propagate in an ideal elastic medium (a string) can be formalised with the D'Alembert equation

$$\frac{\partial^2 y}{\partial x^2}(x, t) = \frac{1}{c^2} \frac{\partial^2 y}{\partial t^2}(x, t)$$

Sometimes, could become useful using the 1D representation for spherical geometry of the D'Alembert equation. Higher-order modes can be neglected, and radial propagation is conveniently described by *zero-order mode*. By applying the Laplacian to the equation, what we get is

$$\frac{1}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial R}{\partial r} \right)(r, t) = \frac{1}{c^2} \frac{\partial^2 R}{\partial t^2}(r, t)$$

After replacing  $R = \frac{\tilde{R}}{r}$  we obtain the 1D D'Alembert equation in  $\tilde{R}$ .



Anyway, to describe a wave that is propagating along a specific direction (planar wave), we prefer to use Cartesian coordinates.

$$\frac{\partial^2 y}{\partial x^2}(x, t) = \frac{1}{c^2} \frac{\partial^2 y}{\partial t^2}(x, t) \rightarrow \underbrace{\left( \frac{\partial}{\partial x} - \frac{1}{c} \frac{\partial}{\partial t} \right)}_{\Delta^+} \underbrace{\left( \frac{\partial}{\partial x} + \frac{1}{c} \frac{\partial}{\partial t} \right)}_{\Delta^-} y = 0 \quad \begin{cases} \Delta^+ y^+ = 0 \\ \Delta^- y^- = 0 \end{cases}$$

In this case, the solution of the D'Alembert equation can be written as the combination of two components, one that solves  $\Delta^+ y^+ = 0$  and a second one that solves  $\Delta^- y^- = 0$ :

$$y(x, t) = y^+(ct - x) + y^-(ct + x)$$

where  $y^+$  describes a planar wave that propagates towards x-increasing coordinates, and  $y^-$  the opposite. As the equation depends both on time and space, we have to provide a condition on time (initial condition) and one on space (boundary condition):

- Initial condition

$$y_0(x) = y(x, 0) \quad v_0(x) = \frac{\partial y}{\partial t}(x, 0)$$

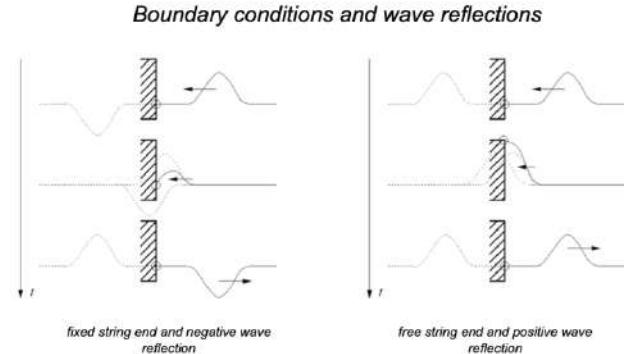
- Boundary condition

Fixed end:

$$y(x, t)|_{x=0} = 0 \rightarrow v_0(x) = \frac{\partial y}{\partial t}(x, 0)$$

Free end:

$$\frac{\partial y}{\partial x}(x, t)|_{x=0} = 0 \rightarrow y^+(ct) = y^-(ct)$$



Let us now consider an ideal string with fixed-end boundary conditions

$$y(x, t)|_{x=0,L} \equiv 0$$

And let us look for stationary waves (that are two solutions, one in function of time-only and one in function of space-only, whose products returns the original expression).

$$y(x, t) = y_1(x)y_2(t) \rightarrow \frac{\partial^2 y}{\partial x^2}(x, t) = \frac{1}{c^2} \frac{\partial^2 y}{\partial t^2}(x, t) \rightarrow \frac{y_1''}{y_1}(x) = \frac{1}{c^2} \frac{\ddot{y}_2}{y_2}(t) \rightarrow \frac{y_1''(x)}{y_1(x)} = \alpha \quad \ddot{y}_2(t) = c^2 \alpha y_2(t)$$

By applying the boundary conditions, we get

$$y_1(x) = \underbrace{\sqrt{\frac{2}{L}}}_{\text{Normalisation factor}} \sin(k_n x), \quad k_n = \frac{n\pi}{L}$$

As we notice, only a numerable set of frequencies is allowed.

By switching to the temporal solution, we get

$$y_2(t) = A \sin(\omega_n t + \phi), \quad \omega_n = ck_n = \frac{n\pi c}{L}$$

The final solution will be expressed by

$$y(x, t) = \sum_{n=1}^{+\infty} A_n y_n(x, t)$$

$$y_n(x, t) = \sqrt{\frac{2}{L}} \sin(\omega_n t + \phi_n) \sin(k_n x)$$

Where  $A_n, \phi_n$  are determined by the initial conditions.

By applying the Werner formula, we obtain

$$y_n(x, t) = \sqrt{\frac{1}{2L}} \{ \cos[k_n(ct - x) + \phi_n] - \cos[k_n(ct + x) + \phi_n] \}$$

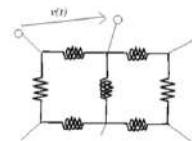
With these passages, we demonstrated that a standing wave is the sum of two sinusoidal waves travelling at speed  $c$  in opposite directions. A remark that we can make is that D'Alembert and Fourier solutions are equivalent.

### Modelling approaches

There are three main approaches that can be used to describe these systems:

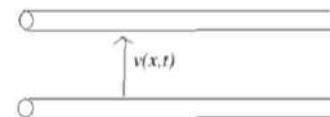
1. Lumped models

The classic one: described by ordinary differential equations (ODEs) based on  $k$  variables or on wave variables (scattering parameters, WDFs). Simply speaking, each element is modelled as an ideal a single device.



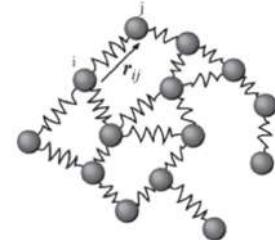
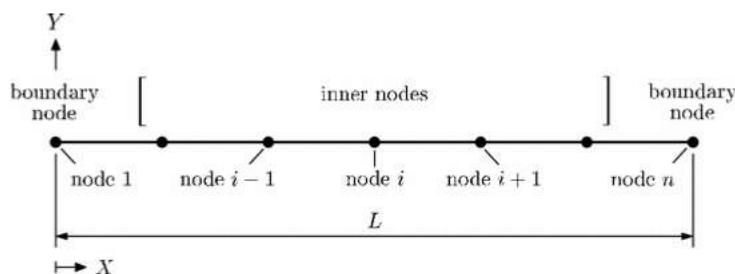
2. Distributed models

Described by Partial Differential Equations PDEs. This technique can be applied discretising the PDEs (with the FD methods, brute force) or discretising the general solution (DWG methods).



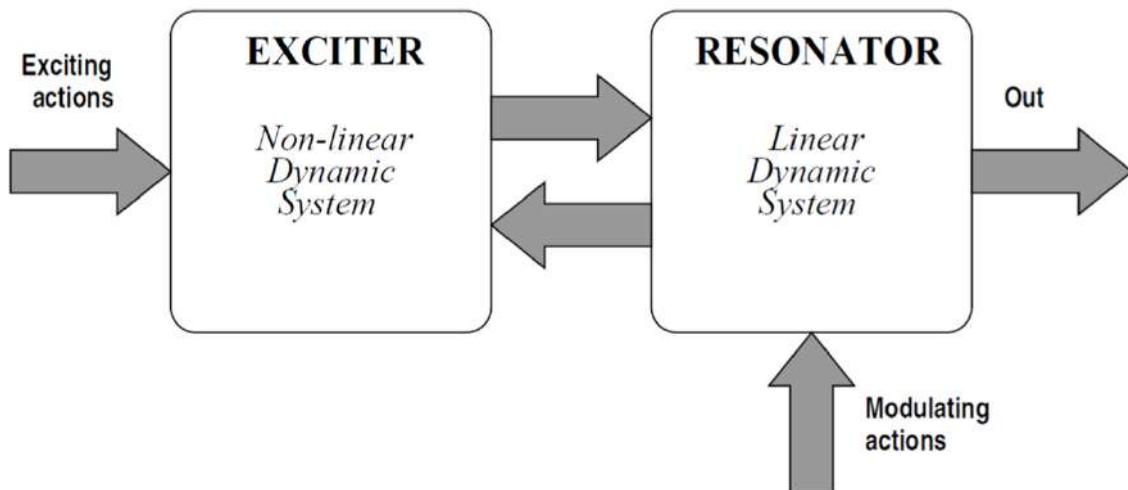
3. Cellular approaches

The model is decomposed into many interacting particles (Cordis-Anima)



### Functional blocks

Every resonating system can be described with two blocks: an exciter (non-linear) and a resonator (LDS) triggered by the exciter itself.

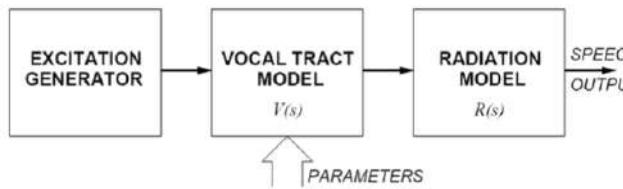


## Feed-forward vs. feedback

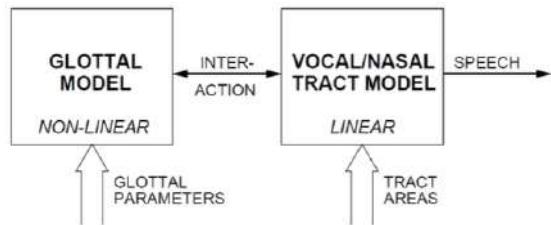
There are two main categories of systems.

Feed Forward:

source-filter scheme of a LP-based speech synthesis



Feedback: articulatory speech synthesiser

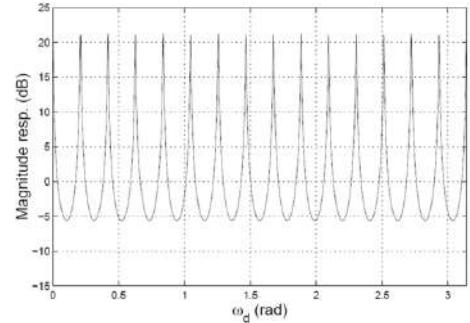
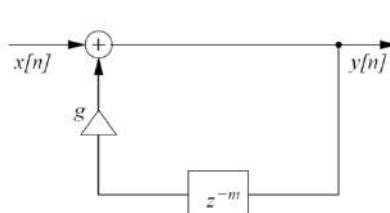
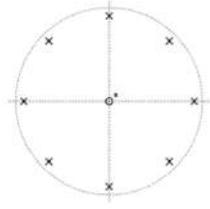


## The Karplus-Strong algorithm

Let us start analysing a first example of system, proposed in the 60s, for modelling the sound of a string. We know that a string oscillates only at multiples of a certain fundamental frequency. The digital system that can emulate this type of system is an IIR comb filter as it generates a series of line equally spaced along the frequency axis.

$$y(n) = x(n) + gy(n-m) \rightarrow Y(z) = X(z) + gz^{-m}Y(z) \rightarrow H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1 - gz^{-m}}$$

poles of  $H(z)$  in  $\{z \in \mathbb{C} | z^m = g\}$



The filter has  $m$  poles in

$$p_l = \sqrt[m]{g} e^{j(2l+1)\frac{\pi}{m}} \quad \text{for } l = 0, \dots, m-1$$

And produces a harmonic spectrum whose frequency peaks are integer multiples of the “fundamental” frequency  $f_0 = \frac{F_s}{m}$  Hz.

If the sign of the wave is inverted at each reflection

$$y(n) = x(n) - gy(n-m) \rightarrow H(z) = \frac{1}{1 + gz^{-m}}$$

Then the poles are:

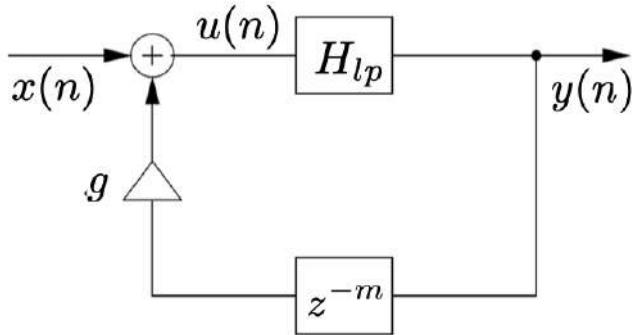
$$p_l = \sqrt[m]{g} e^{j(2l+1)\frac{\pi}{m}} \quad \text{with } l = 0, \dots, m-1$$

Therefore, the frequency peaks are odd integer multiples of the fundamental frequency

$$f_0 = \frac{F_s}{2m} \text{ Hz}$$

Let us see now how to model a plucked string with the KS algorithm. In order to do it, we need to control the spectral tilt of the response and account for different decay rates of partials.

What we want to obtain is an almost perfect harmonic spectrum in which each partial decays at a different rate, with lower partials surviving longer than higher partials. For what it concerns the generation of partials, we said that a comb filter is perfect. But what about the frequency-dependent decay? It is achieved by inserting a LP filter into the feedback loop. Below, the scheme is reported.



The simplest LPF we can use is a 1<sup>st</sup> order FIR whose transfer function is

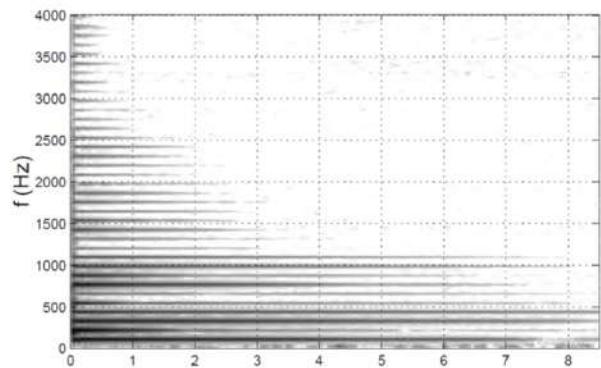
$$y(n) = \frac{1}{2}(u(n) + u(n - 1)) \rightarrow H(z) = \frac{1}{2}(1 + z^{-1})$$

Poles have frequency-dependent magnitudes, as expected. The delay introduced make the fundamental frequency of LP comb structure shift to

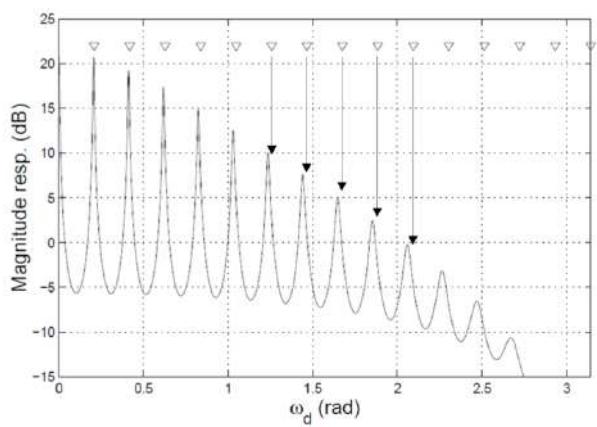
$$f_0 = \frac{F_s}{m + \frac{1}{2}} \text{ Hz}$$

This make upper partials being no longer exact integer multiples of  $f_0$ . As an example, the fundamental of an A4 shifts from 441 Hz to 439 Hz.

What input signal do we feed the filter in order to obtain an output sound? The obvious choice could be the one of injecting the filter with an impulse, as it happens in the physical model (the plucking can be seen as an impulse of energy given to the string). Karplus and Strong suggested to set a random initial state ( $m$  past values of  $y$ ). Despite not having a physical interpretation this solution gives the perceptual effect of an initial noisy transient followed by a harmonic steady-state signal, exactly the one of a string. (The random input generates a significant initial excitation for the high-frequency region as a random signal “changes very fast”). A slightly more in-depth lecture on KS algorithms can be found at <https://youtu.be/WxqsTdxZm8>.



spectrogram of a plucked A2 guitar string



LP comb filter structure: frequency response  
triangles mark harmonic series

If we want to change the fundamental frequency of our system, we just have to change the value of the delay line. In fact, adding a single unit delay in the comb filter shifts the fundamental frequency of

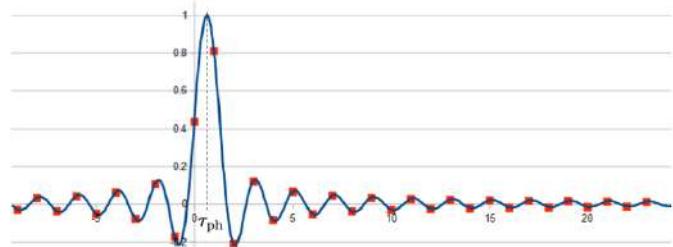
$$\Delta f = \frac{F_s}{M} - \frac{F_s}{M+1} = \frac{F_s}{M(M+1)}$$

Let us make an example with  $F_s = 44 \text{ kHz}$ ,  $M = 100$ ,  $f_0 = \frac{F_s}{M} = 440 \text{ Hz}$ . We get

$$\Delta f = \frac{F_s}{M(M+1)} = 4.36 \text{ Hz} > JND$$

The result does not sound realistic. To fix the problem we work with fine tuning, implementing a fractional delay. The fractional delay can be implemented using a sinc.

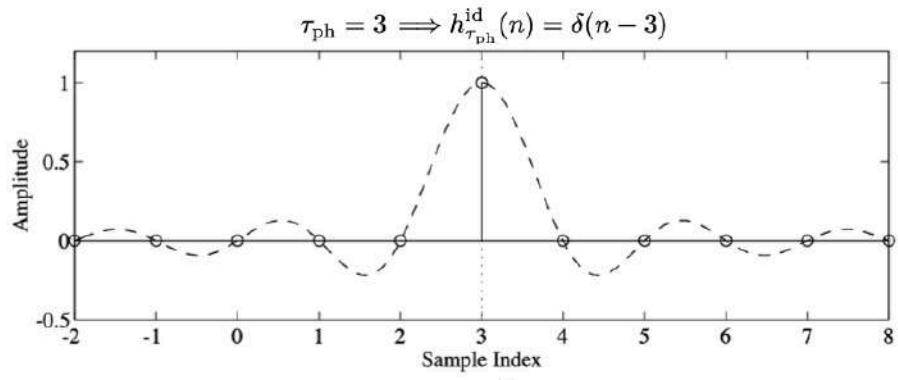
$$y(n) = x(n - \tau_{ph}) = \sum_k x(k) h_{\tau_{ph}}^{id}(n - k)$$



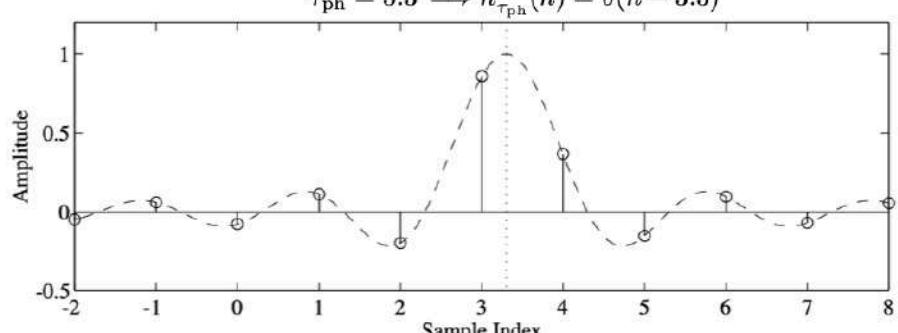
$$H_{\tau_{ph}}^{id}(e^{j\omega}) = e^{-j\omega\tau_{ph}} \quad h_{\tau_{ph}}^{id}(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} e^{-j\omega\tau_{ph}} e^{j\omega n} d\omega = \text{sinc}(n - \tau_{ph})$$

Whenever the delay is an integer number ( $\tau_{ph} = m$ ) the sinc becomes a  $\delta$  ( $h_{\tau_{ph}}^{id}(n) = \delta(n - m)$ ). That's we get.

**Ideal fractional delay filter implementing a delay of 3 samples**



**Ideal fractional delay filter implementing a delay of 3.3 samples**



How do we design a FIR filter (finite) approximating the fractional delay (infinite)?

$$H_{\tau_{ph}}(z) = \sum_{k=0}^N b_k z^{-k}$$

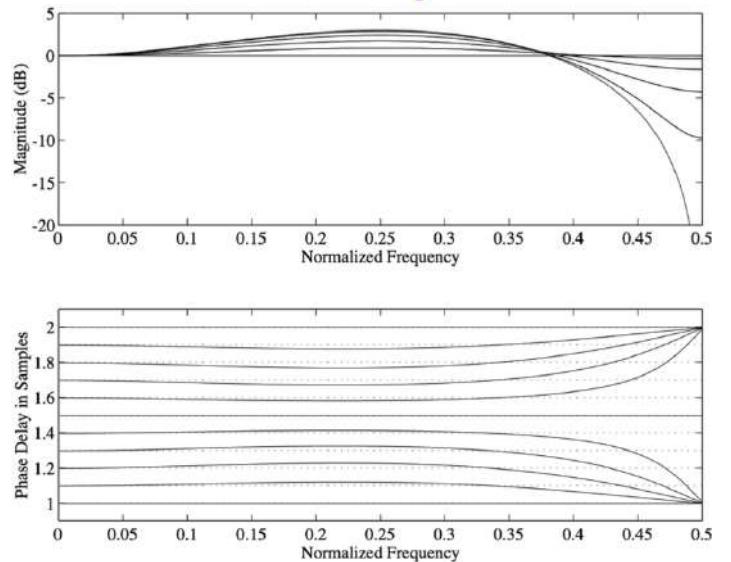
One solution consists of simply truncating the IIR ideal impulse response

$$h_{\tau_{ph}}(n) = \begin{cases} \text{sinc}(n - \tau_{ph}), & 0 \leq n \leq N \\ 0, & \text{otherwise} \end{cases}$$

The delay  $\tau_{ph}$  should be located between the two central taps of the filter when  $N$  is odd ( $L = N + 1$  is even), or within half a sample from the central tap when  $N$  is even ( $L$  odd), in order to minimise the approximation error. This means that the delay should be chosen so that

$$\frac{N-1}{2} \leq \tau_{ph} \leq \frac{N+1}{2}$$

Here are reported the magnitude responses (upper) and phase delay curves (lower) of a 3rd-order FIR filter designed by truncating the shifted sinc function. The curves are plotted for 11 fractional delay values between 1.0 and 2.0. Notice that the magnitude responses for  $N - D$  are the same as for  $D$ . We highlight that, in the phase graph, the case of a delay of 1.5 samples is constant for numerical reasons.



We can mathematically demonstrate that the truncation of the STFT is the best we can obtain. This method minimises the Least Squared (LS) error function  $E_{LS}$ , which is equal to the  $L^2$  norm (integrated squared magnitude) of the error frequency response  $E(e^{j\omega})$

$$E_{LS} = 1 + \sum_{n=0}^N [h_{\tau_{ph}}^2(n) - 2h_{\tau_{ph}}(n) \text{sinc}(n - \tau_{ph})]$$

At the end of the demonstration, we end up with the result that was intuitive at the beginning.

$$h_{\tau_{ph}}(n) = \begin{cases} \text{sinc}(n - \tau_{ph}), & 0 \leq n \leq N \\ 0, & \text{otherwise} \end{cases}$$

With this filter, we had overshoot and undershoot of the magnitude since we are windowing the ideal response with a rectangular window. The rectangular window generates big side lobes. In order to attenuate this phenomenon, we define the impulse response as

$$h_{\tau_{ph}}(n) = \begin{cases} w(n - \tau_{ph}) \text{sinc}(n - \tau_{ph}), & M \leq n \leq M + N \\ 0, & \text{otherwise} \end{cases}$$

Where the midpoint of the window function  $w(n)$  of length  $N + 1$  is shifted by  $\tau_{ph}$  so that the shifted sinc function will be windowed symmetrically with respect to its centre

$$M = \begin{cases} \text{round}(\tau_{ph}) - \frac{N}{2}, & \text{for even } N \\ \lfloor \tau_{ph} \rfloor - \frac{N-1}{2}, & \text{otherwise} \end{cases}$$

### General Least Squares (GLS) Method

Another approach is that of considering just a part of the frequency domain relevant for us in terms of minimisation of the error. This is what is done by using a General Least Squares (GLS) FIR Approximation of a Complex Frequency Response. In principle, the FIR fractional delay filter with the smallest LS error in the defined approximation band is accomplished by defining the response only in that part of the frequency band and by leaving the rest out of the error measure as a “don’t care” band.

$$E_{LS} = \frac{1}{2\pi} \int_{-\alpha\pi}^{\alpha\pi} W(\omega) |E(e^{j\omega})|^2 d\omega = \frac{1}{2\pi} \int_{-\alpha\pi}^{\alpha\pi} W(\omega) |H_{\tau_{ph}}(e^{j\omega}) - H_{\tau_{ph}}^{id}(e^{j\omega})|^2 d\omega$$

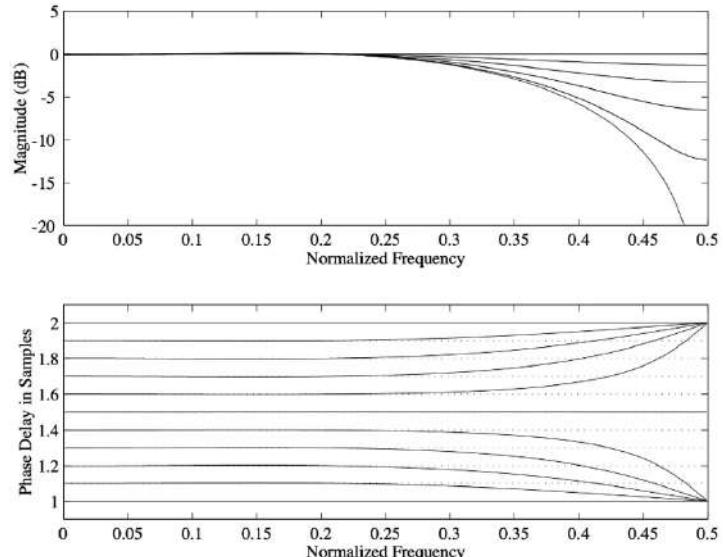
Where the error is defined in the lowpass frequency band  $[-\alpha\pi, \alpha\pi]$

We notice in fact that  $E_{LS}$  is computed between  $-\alpha\pi$  and  $\alpha\pi$ . What it remains is nothing but a don’t care band. Due to this approximation, we can no longer use the Parseval’s theorem.

On the right, are reported the magnitude responses (upper) and the phase delay curves (lower) of a 3<sup>rd</sup>-order GLS FIR filter with  $\alpha = 0.5$

Curves are plotted for 11 fractional delay values between 1.0 and 2.0

We can notice that the magnitude responses for  $N - D$  are the same as for  $D$ .



### Maximally-Flat Fractional Delay Filter

Another possible solution is the so-called maximally flat design. Instead of working with the whole spectrum, this solution imposes to have a response that has l-order derivative = 0 at the frequency  $\omega = 0$ . So, we consist of setting the error function

$$E(e^{j\omega}) = H_{\tau_{ph}}(e^{j\omega}) - e^{-j\omega\tau_{ph}}$$

And its first  $N$  derivatives to zero at a given frequency, which we conveniently assume to be  $\omega = 0$

$$\left. \frac{d^l E}{d\omega^l}(e^{j\omega}) \right|_{\omega=0} = 0 \quad l = 0, \dots, N$$

By substituting  $E(e^{j\omega})$  we get

$$\frac{d^l H_{\tau_{ph}}}{d\omega^l}(e^{j\omega}) \Big|_{\omega=0} = \frac{d^l(e^{-j\omega\tau_{ph}})}{d\omega^l} \Big|_{\omega=0} \quad l = 0, \dots, N$$

Let us now see the first terms of the sum

$l = 0$

$$\sum_{n=0}^N h_{\tau_{ph}}(n) e^{-j\omega n} \Big|_{\omega=0} = e^{-j\omega\tau_{ph}} \Big|_{\omega=0} \rightarrow \sum_{n=0}^N h_{\tau_{ph}}(n) = 1$$

$l = 1$

$$\frac{d}{d\omega} \sum_{n=0}^N h_{\tau_{ph}}(n) e^{-j\omega n} \Big|_{\omega=0} = \frac{d(e^{-j\omega\tau_{ph}})}{d\omega} \Big|_{\omega=0} \rightarrow -j \sum_{n=0}^N n h_{\tau_{ph}}(n) = -j\tau_{ph} \rightarrow \sum_{n=0}^N n h_{\tau_{ph}}(n) = \tau_{ph}$$

$l = 2$

$$\frac{d^2}{d\omega^2} \sum_{n=0}^N h_{\tau_{ph}}(n) e^{-j\omega n} \Big|_{\omega=0} = \frac{d^2(e^{-j\omega\tau_{ph}})}{d\omega^2} \Big|_{\omega=0} \rightarrow \sum_{n=0}^N n^2 h_{\tau_{ph}}(n) = \tau_{ph}^2 \rightarrow$$

Any  $l$

$$\frac{d^l}{d\omega^l} \sum_{n=0}^N h_{\tau_{ph}}(n) e^{-j\omega n} \Big|_{\omega=0} = \frac{d^l(e^{-j\omega\tau_{ph}})}{d\omega^l} \Big|_{\omega=0} \rightarrow \sum_{n=0}^N n^l h_{\tau_{ph}}(n) = \tau_{ph}^l \rightarrow$$

So what we get is a set of  $N + 1$  linear equations

$$\sum_{n=0}^N n^l h_{\tau_{ph}}(n) = \tau_{ph}^l \quad l = 0, 1, \dots, N$$

That can be re-written in a matrix fashion as

$$\underline{\underline{V}} \underline{h} = \underline{v}$$

where  $\underline{\underline{V}}$  is a  $(N + 1) \times (N + 1)$  Vandermonde matrix (which is a non-singular matrix)

$$\underline{\underline{V}} = \begin{pmatrix} 0^0 & 1^0 & 2^0 & \dots & N^0 \\ 0^1 & 1^1 & 2^1 & \dots & N^1 \\ 0^2 & 1^2 & 2^2 & \dots & N^2 \\ \vdots & & & \ddots & \vdots \\ 0^N & 1^N & 2^N & \dots & N^N \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & \dots & 1 \\ 0 & 1 & 2 & \dots & N \\ 0 & 1 & 4 & \dots & N^2 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & 1 & 2^N & \dots & N^N \end{pmatrix}, \quad \underline{h} = \begin{pmatrix} h_{\tau_{ph}}(0) \\ h_{\tau_{ph}}(1) \\ h_{\tau_{ph}}(2) \\ \vdots \\ h_{\tau_{ph}}(N) \end{pmatrix}, \quad \underline{v} = \begin{pmatrix} 1 \\ \tau_{ph} \\ \tau_{ph}^2 \\ \vdots \\ \tau_{ph}^N \end{pmatrix}$$

$\underline{\underline{V}}$  is a non-singular matrix therefore the “maximally flat” DF filter can be found as

$$\underline{h} = \underline{\underline{V}}^{-1} \underline{v}$$

where  $\underline{\underline{V}}^{-1}$  can be computed using Cramer’s rule.

Reminder: Cramer’s rule for inverting matrixes

$$\underline{\underline{A}}^{-1} = \frac{1}{\det(\underline{\underline{A}})} \begin{pmatrix} \underline{\underline{A}}_{11} & \underline{\underline{A}}_{21} & \cdots & \underline{\underline{A}}_{n1} \\ \underline{\underline{A}}_{12} & \underline{\underline{A}}_{22} & \cdots & \underline{\underline{A}}_{m2} \\ \vdots & \cdots & \ddots & \vdots \\ \underline{\underline{A}}_{1n} & \underline{\underline{A}}_{2n} & \cdots & \underline{\underline{A}}_{nn} \end{pmatrix}$$

Example: find the inverse, if it exists, for

$$\underline{\underline{A}} = \begin{pmatrix} 0 & 1 & 2 \\ -2 & 3 & -1 \\ 4 & 0 & 1 \end{pmatrix}$$

We find the submatrix by deleting the row and the column identified by the subscripts:

$$\underline{\underline{A}}_{11} = \begin{vmatrix} 3 & -1 \\ 0 & 1 \end{vmatrix} = 3, \quad \underline{\underline{A}}_{12} = \begin{vmatrix} -2 & -1 \\ 4 & 1 \end{vmatrix} = -2, \quad \underline{\underline{A}}_{13} = \begin{vmatrix} -2 & 3 \\ 4 & 0 \end{vmatrix} = -12$$

Find the determinant by the expansions along the first row:

$$\det(\underline{\underline{A}}) = a_{11}\underline{\underline{A}}_{11} + a_{12}\underline{\underline{A}}_{12} + a_{13}\underline{\underline{A}}_{13} = 0 \cdot 3 + 1 \cdot (-2) + 2 \cdot (-12) = -26$$

Since  $\det(\underline{\underline{A}}) \neq 0$ , we conclude that  $\underline{\underline{A}}$  is invertible, and we can continue computing cofactors. If the determinant were 0, we would stop here and say that  $A$  is singular (there is no need to find rest cofactors).:

$$\begin{aligned} \underline{\underline{A}}_{21} &= - \begin{vmatrix} 1 & 2 \\ 0 & 1 \end{vmatrix} = -1, & \underline{\underline{A}}_{22} &= \begin{vmatrix} 0 & 2 \\ 4 & 1 \end{vmatrix} = -8, & \underline{\underline{A}}_{23} &= - \begin{vmatrix} 0 & 1 \\ 4 & 0 \end{vmatrix} = 4 \\ \underline{\underline{A}}_{31} &= \begin{vmatrix} 1 & 2 \\ 3 & -1 \end{vmatrix} = -7, & \underline{\underline{A}}_{32} &= - \begin{vmatrix} 0 & 2 \\ -2 & -1 \end{vmatrix} = 4, & \underline{\underline{A}}_{33} &= \begin{vmatrix} 0 & 1 \\ -2 & 3 \end{vmatrix} = 2 \end{aligned}$$

By using the formula above we get

$$\underline{\underline{A}}^{-1} = -\frac{1}{26} \begin{pmatrix} 3 & -1 & -7 \\ -2 & -8 & -4 \\ -12 & 4 & 2 \end{pmatrix} = \begin{pmatrix} -\frac{3}{26} & \frac{1}{26} & \frac{7}{26} \\ \frac{1}{13} & \frac{4}{13} & \frac{2}{13} \\ \frac{6}{13} & -\frac{2}{13} & \frac{1}{13} \end{pmatrix}$$

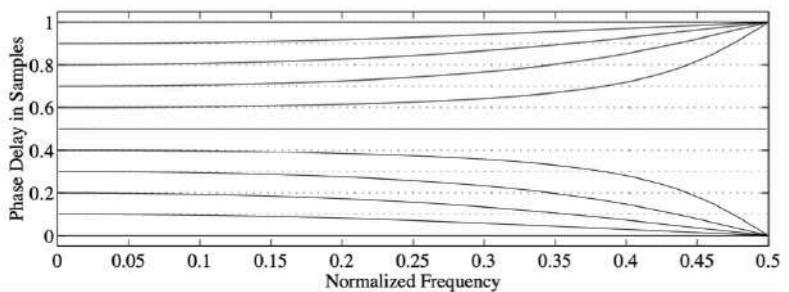
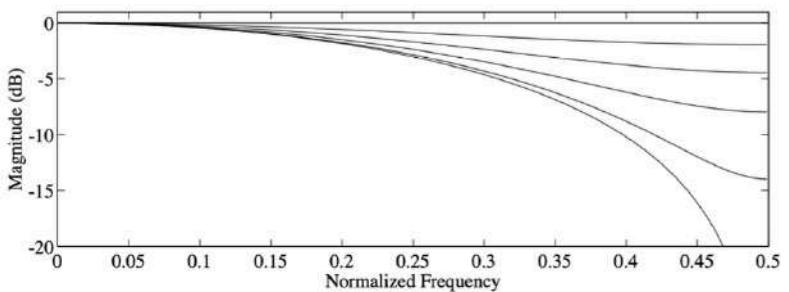
As a test, we can see if  $\underline{\underline{A}} \underline{\underline{A}}^{-1} = \underline{\underline{I}}$

$N = 1$ 

Magnitude (upper) and phase delay (lower) response of a linear interpolator for  $11$  different fractional delay values ( $0, 0.1, 0.2, \dots, 1.0$ )

Only six curves are shown in the upper figure as the magnitude responses for fractional delays  $d$  and  $1-d$  are the same.

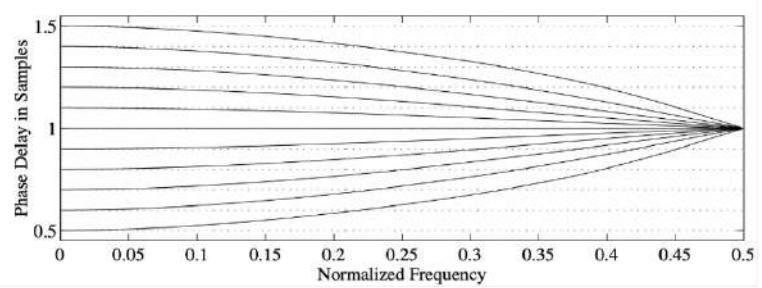
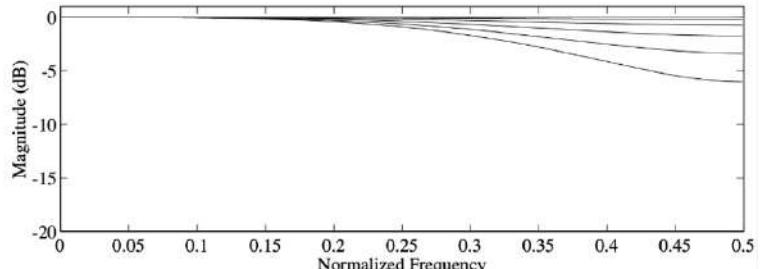
In the lower figure, the ideal phase delay in each case is shown by a dotted line

 $N = 2$ 

Magnitude (upper) and phase delay (lower) response of a 2nd-order interpolator for  $11$  different fractional delay values ( $0.5, 0.6, \dots, 1.4, 1.5$ )

Only six curves are shown in the upper figure as the magnitude responses for fractional delays  $d$  and  $N-d$  are the same.

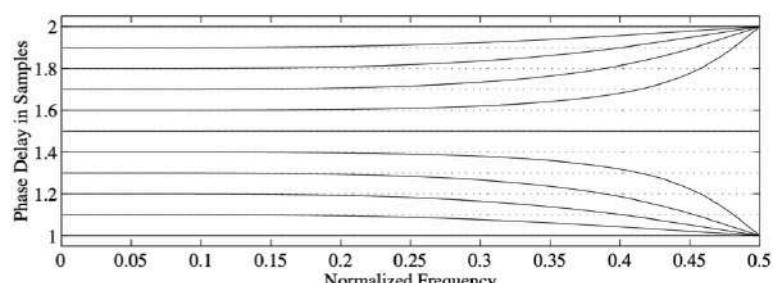
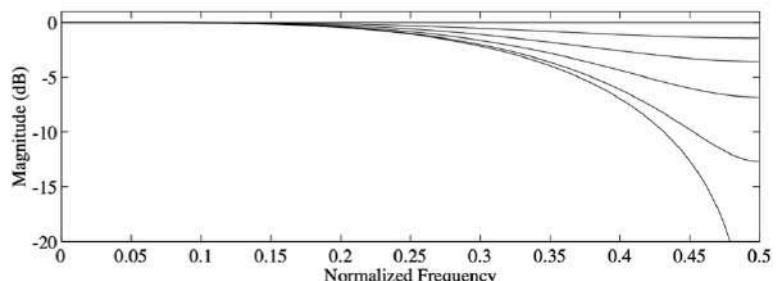
In the lower figure, the ideal phase delay in each case is shown by a dotted line

 $N = 3$ 

Magnitude (upper) and phase delay (lower) response of a 3rd-order interpolator for  $11$  different fractional delay values ( $1.0, 1.1, 1.2, \dots, 2.0$ )

Only six curves are shown in the upper figure as the magnitude responses for fractional delays  $d$  and  $N-d$  are the same.

In the lower figure, the ideal phase delay in each case is shown by a dotted line.

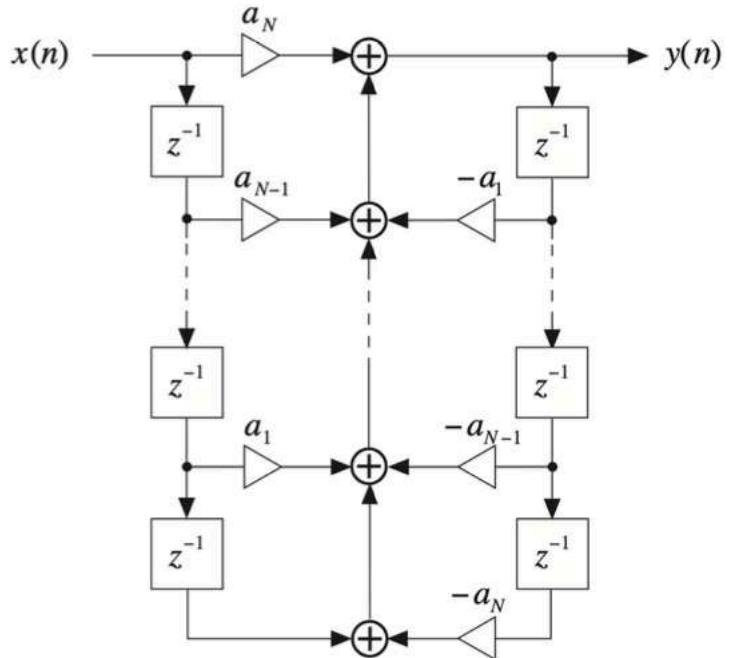


## All-pass fractional delay filters

How about using an IIR filter as an all-pass filter?

So far, we have seen how to design FIR filters for fractional delay approximation. Now we show how IIR filters can be used for the same design problem:

- As the magnitude response of an ideal FD element is perfectly flat, we only consider allpass filters, because their magnitude response is always flat irrespective of the filter coefficients
- Allpass filters are typically used for phase equalization and other signal processing tasks where the phase characteristics are of greatest concern
- The Fractional Delay approximation is essentially a phase approximation problem



$$H_{\tau_{ph}}(z) = \frac{z^{-N} A(z)}{A(z^{-1})} = \frac{a_n + a_{n-1}z^{-1} + \dots + a_1z^{-(N-1)} + z^{-N}}{1 + a_1z^{-1} + \dots + a_{N-1}z^{-(N-1)} + a_Nz^{-N}}$$

The numerator of the all-pass transfer function is a mirrored version of the denominator polynomial. Poles are located inside the unit circle in the complex plane (stability). Therefore, zeros must be located outside. Consequently, the magnitude response of an all-pass filter is flat.

$$\left| H_{\tau_{ph}}(e^{j\omega}) \right| = \left| e^{-j\omega N} \frac{A(e^{-j\omega})}{A(e^{j\omega})} \right| \equiv 1$$

We, therefore, can write

$$H_{\tau_{ph}}(e^{j\omega}) = e^{j\Theta_H(\omega)}$$

As far as phase response is concerned, we have

$$\Theta_H(\omega) = \arg \{ H_{\tau_{ph}}(e^{j\omega}) \} = -N\omega + 2\Theta_A(\omega)$$

where  $\Theta_A(\omega)$  is the phase response of  $A(\omega)$ . If we know how to design the phase response of an all-pole (AR) filter, we know how to design the corresponding all-pass filter  $H_{\tau_{ph}}(z)$  using  $\Theta_H(\omega)$ .

As we said,  $\Theta_A(\omega)$  is the phase response of  $A(\omega)$

$$\Theta_A(\omega) = \arg\{A(e^{j\omega})\} = \arctan \left\{ \frac{\sum_{k=0}^N a_k \sin(k\omega)}{\sum_{k=0}^N a_k \cos(k\omega)} \right\} = \arctan \left\{ \frac{\underline{a}^T \underline{s}}{\underline{a}^T \underline{c}} \right\}$$

where

$$\underline{a} = \begin{pmatrix} 1 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{pmatrix}, \quad \underline{s} = \begin{pmatrix} 0 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{pmatrix}, \quad \underline{c} = \begin{pmatrix} 1 \\ \cos(\omega) \\ \cos(2\omega) \\ \vdots \\ \cos(N\omega) \end{pmatrix}$$

Recalling that

$$\Theta_H(\omega) = -N\omega + 2\Theta_A(e^{j\omega})$$

we have

$$\Theta_H(\omega) = -N\omega + 2 \arctan \left\{ \frac{\underline{a}^T \underline{s}}{\underline{a}^T \underline{c}} \right\}$$

The corresponding phase delay can therefore be derived as

$$\tau_{p,H}(\omega) = -\frac{\Theta_H(\omega)}{\omega} = N - 2\tau_{p,A}(\omega)$$

while the corresponding group delay is

$$\tau_{g,H}(\omega) = -\frac{d\Theta_H(\omega)}{d\omega} = N - 2\tau_{g,A}(\omega)$$

There is a great difference between the design of FIR and all-pass filters. The coefficients of an FIR filter are easily obtained as the inverse DTFT of the freq.-domain specs. The relationship between the coefficients of the all-pass transfer function and the samples of the corresponding impulse response is not that simple. Most of the design techniques for all-pass filters are iterative.

The desired or ideal phase response in the fractional delay approximation problem is

$$\Theta_{id}(\omega) = -\tau_{ph}\omega$$

The phase error of an all-pass filter can be defined as the deviation from the desired phase function. The phase error of an all-pass filter can be defined as the deviation from the desired phase function.

$$\Delta\Theta = \Theta_{id}(\omega) - \Theta_H(\omega) = \Theta_{id}(\omega) + N\omega - 2 \arctan \left\{ \frac{\underline{a}^T \underline{s}}{\underline{a}^T \underline{c}} \right\} = 2 \arctan \left\{ \frac{\underline{a}^T \underline{s}_\beta}{\underline{a}^T \underline{c}_\beta} \right\}$$

The things explained in this lesson are exactly the same analysed by Oliviero Massi during the 16<sup>th</sup> April's lesson. For this reason, we skip them.

### Discretisation of Lumped models

We know that a differential equation can be properly and simply described by its Laplace transform.

$$\Gamma(s) = \frac{B(s)}{A(s)} = \frac{\sum_{k=0}^M b_k s^{M-k}}{\sum_{k=0}^N a_k s^{N-k}} \quad \rightarrow \quad \Gamma(s) = \sum_{k=1}^N \frac{K_k}{s - p_k}$$

The form

$$\gamma_d[n] \triangleq \gamma(nT_s) = \sum_{k=1}^N K_k (e^{p_k T_s})^n$$

is quite convenient since we make explicit the dependence from the exponent  $n$ . The Z-transform will be therefore represented by

$$\Gamma_d(z) = \sum_{k=1}^N \frac{K_k}{1 - p_{d,k} z^{-1}} = \frac{B_d(z)}{A_d(z)} \quad \text{with } p_{d,k} = e^{p_k T_s}$$

The transfer function of the discretised system is still rational. Its  $N$  poles can be derived from the poles of the analogical system. The dissertation scheme is called impulse invariant-discretisation (simply speaking, a sampling of the impulse response).

Let us list some considerations:

1. Stability is guaranteed at any sampling rate: if the continuous-time system is stable, so is the discrete-time system.
2. The method suffers from aliasing phenomena: the discrete-time response is a periodisation of the continuous-time response

$$\Gamma_d(e^{j\omega}) = \sum_{k=-\infty}^{\infty} \Gamma\left(\frac{j\omega}{T_s} + j\frac{2k\pi}{T_s}\right)$$

Aliasing occurs if the bandwidth exceeds the Nyquist frequency

An alternate approach can be found by replacing time derivatives with finite differences and turn ODE into difference equations:

- CT derivative corresponds to a multiplication by  $s$
- DT unit delay corresponds to a multiplication by  $z^{-1}$

We notice how approximating derivatives with FDs corresponds to finding an appropriate s-to-z mapping  $s = g(z)$

$$\Gamma_d(z) = \Gamma(g(z))$$

Let us see an example on how derivative can be seen as incremental ratio

$$\frac{dx}{dt}(nT_s) = \lim_{h \rightarrow 0^+} \frac{x(nT_s) - x(nT_s - h)}{h} \approx \frac{x[n] - x[n-1]}{T_s} \rightarrow s \approx \frac{1 - z^{-1}}{T_s} \triangleq g_1(z)$$

$$\frac{d^2x}{dt^2}(nT_s) \approx \frac{1}{T_s} \left[ \frac{x[n] - x[n-1]}{T_s} - \frac{x[n-1] - x[n-2]}{T_s} \right] = \frac{x[n] - 2x[n-1] + x[n-2]}{T_s^2}$$

By substituting  $n$  with  $n + 1$  we get

$$\frac{d^2}{dt^2}x(t_n) \approx \frac{x[n+1] - 2x[n] + x[n-1]}{T_s^2}$$

That is called central difference

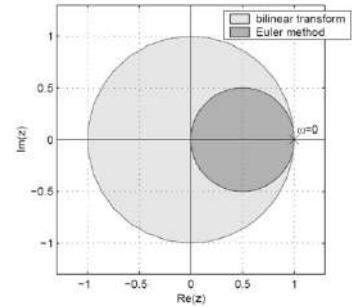
Which is the problem here? In continuous time domain we evaluate the derivative at  $nT_s$  while in the discrete time domain we evaluate it  $n - \frac{1}{2}$  since we are considering  $n$  and  $n - 1$ . In order to evaluate it in  $n$  we shoul evaluate it between  $n + \frac{1}{2}$  and  $n - \frac{1}{2}$ . That is why it is called back-ward method as we look behind for  $\alpha$  samples. There is this misalignment of  $\alpha$  samples.

The expression will become

$$\frac{SX(z)(1 + z^{-1})}{2} = X(z) \frac{(1 - z^{-1})}{T_s} \Rightarrow S = \frac{2(1 - z^{-1})}{T_s(1 + z^{-1})}$$

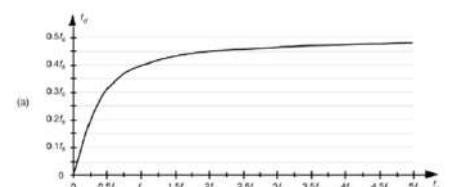
That is called bi-linear transform. We bump into a problem since we have something at the denominator.

The Euler method is stable only in the dark grey region while the bilinear is stable in the standard region.

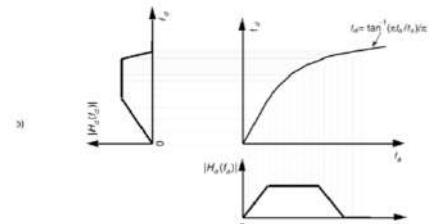
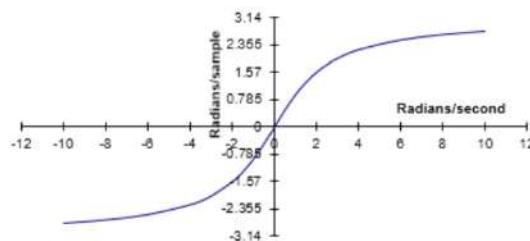


The problem is that with the latter we introduce frequency warping

$$\omega_d = 2 \arctan\left(\frac{\omega T_s}{2}\right)$$



How can we address the problem of frequency warping in the context of bilinear transform? By decreasing the sampling period  $T_s$  so that we are in the area in which the  $\arctan()$  function is linear.



This solution comes at a cost: we have to sample at a higher rate.

Let us now apply these models to real cases:

### String

In this case we have a partial differential equation: the displacement depends on both the position and the time instant. In the simplest case, it can be written as

$$Ty'' = \mu\ddot{y}$$

Where:

- Tension (constant)  $T$
- Linear mass density (constant)  $\mu$
- Displacement  $y \triangleq y(t, x)$
- Velocity  $\dot{y} \triangleq \frac{\partial}{\partial t} y(t, x)$
- Acceleration  $\ddot{y} \triangleq \frac{\partial^2}{\partial t^2} y(t, x)$
- Slope  $y' \triangleq \frac{\partial}{\partial x} y(t, x)$
- Curvature  $y'' \triangleq \frac{\partial^2}{\partial x^2} y(t, x)$
- Propagation velocity  $c = \sqrt{\frac{T}{\mu}}$

Forward (+), Central (+-) and Backward (-) difference approximations will be given by

$$f'(x) \approx \frac{f(x + X_s) - f(x)}{X_s} \quad f'(x) \approx \frac{f(x + X_s) - f(x - X_s)}{2X_s} \quad f'(x) \approx \frac{f(x) - f(x - X_s)}{X_s}$$

Using forward and backward difference approximations for the 2nd partial derivatives and using indices  $n$  and  $k$  for the temporal and spatial points, respectively, yields:

$$\frac{1}{T_s^2} [y(n+1, k) - 2y(n, k) + y(n-1, k)] = \frac{c^2}{X_s^2} [y(n, k-2) - 2y(n, k) + y(n, k-1)]$$

where  $T_s$  and  $X_s$  are the temporal and spatial sampling steps, respectively, and  $c = \sqrt{\frac{T}{\mu}}$  is the propagation velocity of the transversal wave.

### Courant-Friedrichs-Lowy condition

Let us consider the comparison between the element that we have on the left of the equation and the one that we have on the right. There are two constrains that have to be satisfied: let us consider two-time instants  $k$  and  $k + 1$ . Between these two time instants, the wave will propagate from the position  $n$  to the position  $n + cT_s$ . But we have to be sure that the distance between the two points is exactly one sample in the space domain otherwise we are subject to a problem similar to aliasing. We have therefore to impose a constrain: the waves must not propagate faster than one spatial interval at each time step. Formally

$$R = \frac{X_s}{cT_s} \leq 1$$

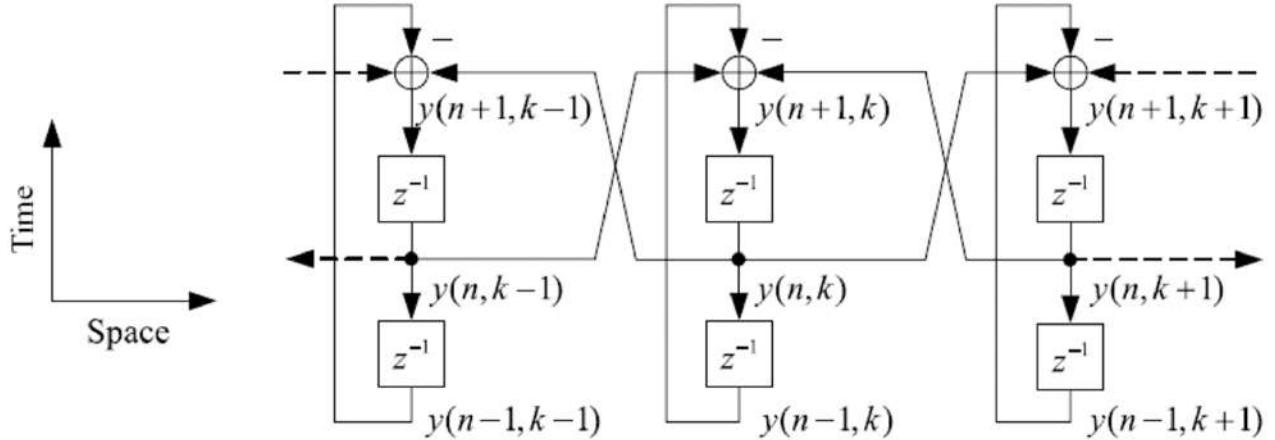
To re-write the wave equation in a computable form we set

$$R = \frac{X_s}{cT_s} = 1 \Leftrightarrow X_s = cT_s$$

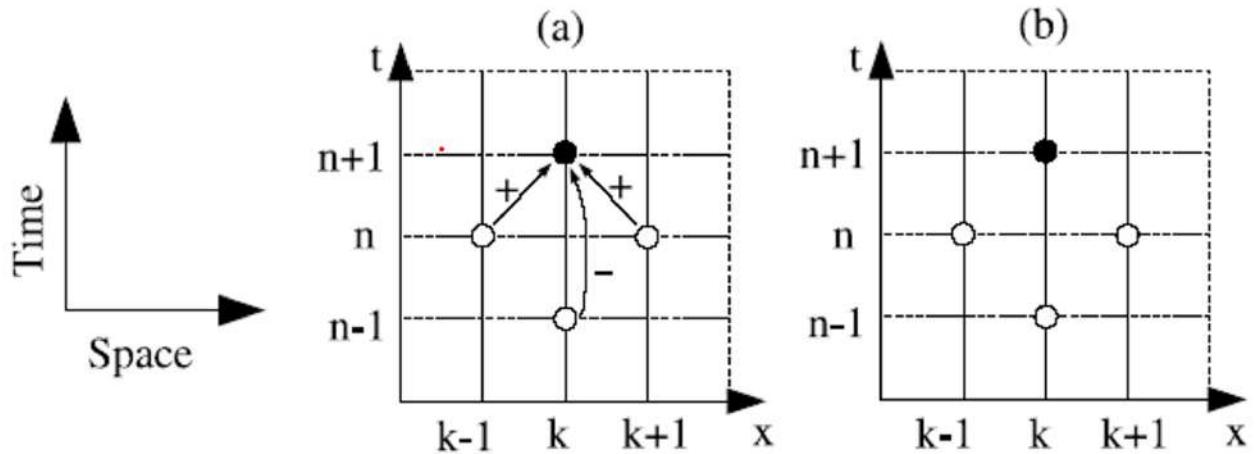
In this way, we can write the wave equation in a computable form

$$y(n+1, k) = y(n, k+1) + y(n, k-1) - y(n-1, k)$$

This condition is called Courant-Friedrichs-Lowy condition. It can be modelled with the following computational scheme.



The kind of scheme we see is called Leapfrog recursion (as it seems the path of a frog that jumps hither and thither). It could be convenient to represent the equation with a different type of scheme where horizontal axis represents the spatial coordinate (i.e. position on the string) and the vertical axis represents the temporal coordinate (i.e. the time instant).



$$y(n+1, k) = y(n, k+1) + y(n, k-1) - y(n-1, k)$$

Ideally, the effect of the boundary condition is negligible. If we want to model it properly, we can represent it with a lossy termination. In this way we have modelled the boundary conditions.

At the position 0

$$y(n+1, 0) = (1 - r_l)y(n, 1) + r_l y(n-1, 0)$$

At the end of the string  $L_{nom}$

$$y(n+1, L_{nom}) = (1 - r_r)y(n, L_{nom} - 1) + r_r y(n-1, L_{nom})$$

Where  $r_l$  and  $r_r$  are two coefficients used to model the losses.

What about the excitation?

A string can be excited in many ways. We see now how to model a plucked string. The simplest solution could be the one to set the initial shape to match the displaced string prior to the release and then carry on with the simulation. For example, by defining a triangular initial shape with no initial velocity.

$$y(1, k) = \begin{cases} y(0, k), & k = 0, 1, \dots, L_{nom} \quad \text{with } k \neq j, j + 1 \\ \frac{1}{2}[y(0, k - 1) + y(0, k + 1)], & k = j, j + 1 \end{cases}$$

A smarter solution enables the interaction with the string during run-time. Here, an external excitation signal  $u$  is inserted into the string ('boxcar' block function), which spreads in both directions from the excitation point pair.

$$y(n, k) \leftarrow y(n, k) + \frac{1}{2}u(n) \quad y(n, k + 1) \leftarrow y(n, k + 1) + \frac{1}{2}u(n)$$

### FD approximation of a lossy string

Up to now we have not considered losses inside the string. In some applications (for example when we use gut strings) these losses are not negligible at all. Let us see how to include them. The equation becomes

$$Ty'' = \mu\ddot{y} + d_1\dot{y} - d_2\dot{y}''$$

where  $d_1$  and  $d_2$  are coefficients that represent the frequency-independent and frequency-dependent damping, respectively. The last term is a mixed time-space derivative instead of a third-order time derivative because the third-order time derivative term tends to make the system ill-posed and can lead to numerical instability if the temporal and spatial sampling frequencies are high enough.

This continuous equation can be translated into a discrete one with the following formula.

$$y(n + 1, k) = g_k[y(n, k + 1) + y(n, k - 1)] - a_k y(n - 1, k)$$

with

$$g_k = \frac{1}{1 + d_1 \frac{T_s}{2}} \quad a_k = \frac{1 - d_1 \frac{T_s}{2}}{1 + d_1 \frac{T_s}{2}}$$

The string decay time can now be directly controlled via the damping coefficient  $d_1$ , since  $d_1 = \frac{1}{2\tau}$ , where  $\tau$  is the time constant for which the amplitude of the vibration decays to  $\frac{1}{e}$  of its initial value.

## Stiffness in FS strings

Whenever we have a string that has a certain stiffness, we have to take into account also the dispersion. That is why a piano tuner changes the fundamental frequency of some strings in order not to create problems with the partials. A model that takes into account also the stiffness is given by

$$Ty'' = \mu \ddot{y} + Dy''' \quad \text{with } D = \frac{E\pi r_s^4}{4}$$

Where  $E$  is Young's modulus and  $r_s$  is the cross-section radius of the string.

## Cellular approach (*it is out of the program for the exam*)

Unlike FD methods, in cellular methods atomization (divide the system into cells) precedes discretization! The physical system to be modelled is approximated by an interconnection of lumped mass elements connected by springs and dampers (links). The network of masses, connected by links, approximates the behaviour of a distributed system. Mass elements correspond to the nodes of a network and are the smallest modules representing inertia. Links represent physical interactions between them. The description is based on dual-pairs of K-variables (position-velocity or force-velocity) | Each element is «locally» discretized (element by element) using FD. The inevitable delay-free loops are forcibly removed by adding a delay element (this

forces causality and creates an artificial order of the operations) the delay causes a two-way interaction to be computed in an «interleaved» fashion (update the status of one mass, then update the other). The fundamental difference between FD schemes and mass–spring networks techniques is that a network is built by connecting modules, rather than by discretizing equations

## Example

Let us analyse the following example.

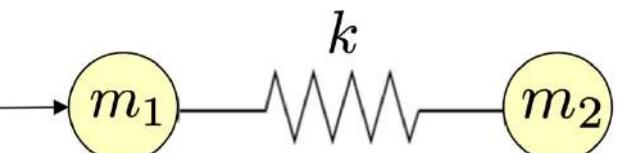
Consider a very simple 1D mechanical system made of two particles  $M_1$  and  $M_2$ , together with an ideal spring  $S_3$  linking them.

The particles move along the  $x$  axis and  $M_1$  is applied an external force  $f_{ext,1}$ .

Let us list now some relevant attributed of the mass and of the spring

Attributes of the mass

- Mass  $m$
- Displacement  $x(t)$
- Velocity  $v(t)$
- Acceleration  $a(t)$
- Total force  $f(t)$  acting on the particle



Attributes of the spring

- Spring constant  $k$
- Length at rest  $l^0$
- Length under tension  $l(t)$

The masses can be described by the Newton's equation

- $f_1(t) = m_1 a_1(t) = f_1 \ddot{x}_1(t)$
- $f_2(t) = m_2 a_2(t) = f_2 \ddot{x}_2(t)$

and the spring by the Hooke's law

- $f_3(t) = k(l_3^0 - l_3(t))$

At this point, we have to find a way to “connect” the equations as we connect the wires on a circuit, that is analysed by using the Kirchhoff's laws. Here we have:

- Spring length as a function of masses displacements (Kirchhoff law of currents)

$$l_3(t) = x_2(t) - x_1(t)$$

- Force equilibria on the masses

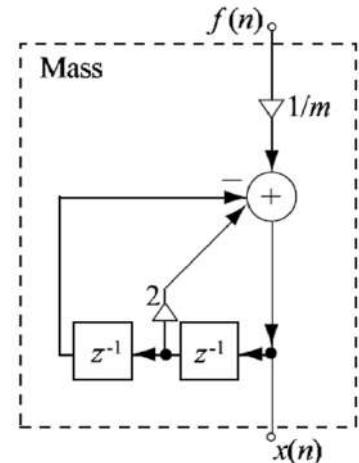
$$f_1(t) = f(t) - f_3(t) \quad \text{and} \quad f_2(t) = f_3(t)$$

We apply now the discretisation in time domain noticing that the system is atomised as it is (the objects cannot be further reduced to springs, dampers and masses because they still are springs, dampers and masses).

$$f_1(t) = m_1 \ddot{x}_1(t) \rightarrow x_1(n) = \frac{f_1(n)}{m_1} + 2x_1(n-1) - x_1(n-2)$$

$$f_2(t) = m_2 \ddot{x}_2(t) \rightarrow x_2(n) = \frac{f_2(n)}{m_2} + 2x_2(n-1) - x_2(n-2)$$

The mass can be represented with one-port element like the one on the right.

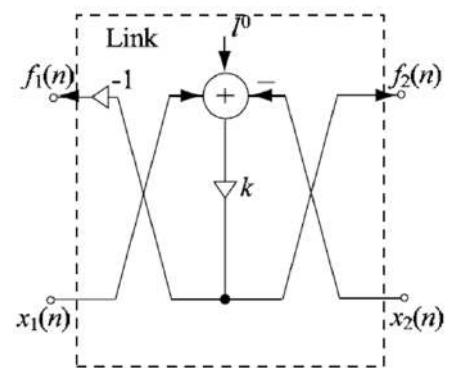


The spring is what connects the two masses. Therefore, it can be seen as a two-port element between them that implements the interaction relations or that computes the output forces as a function of the input displacement.

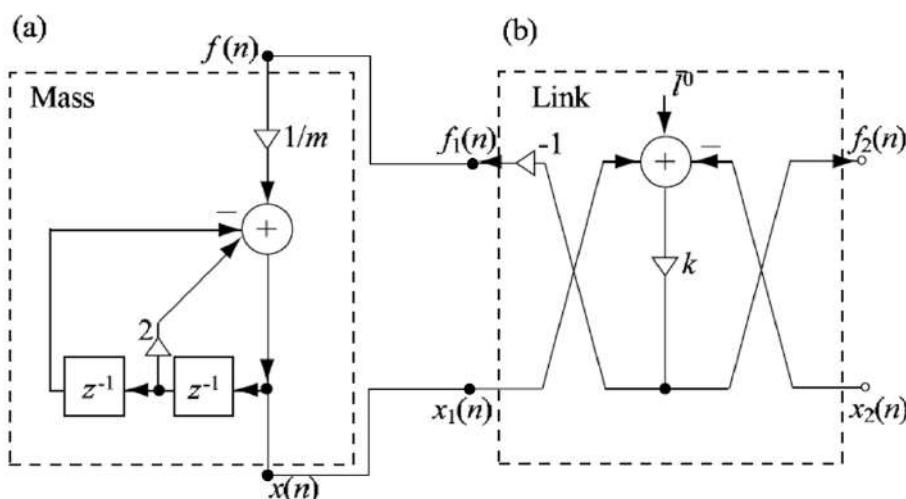
$$f_3(n) = k(l_3^0 - x_2(n) + x_1(n))$$

$$f_1(t) = f - f_3(n)$$

$$f_2(t) = f_3(n)$$

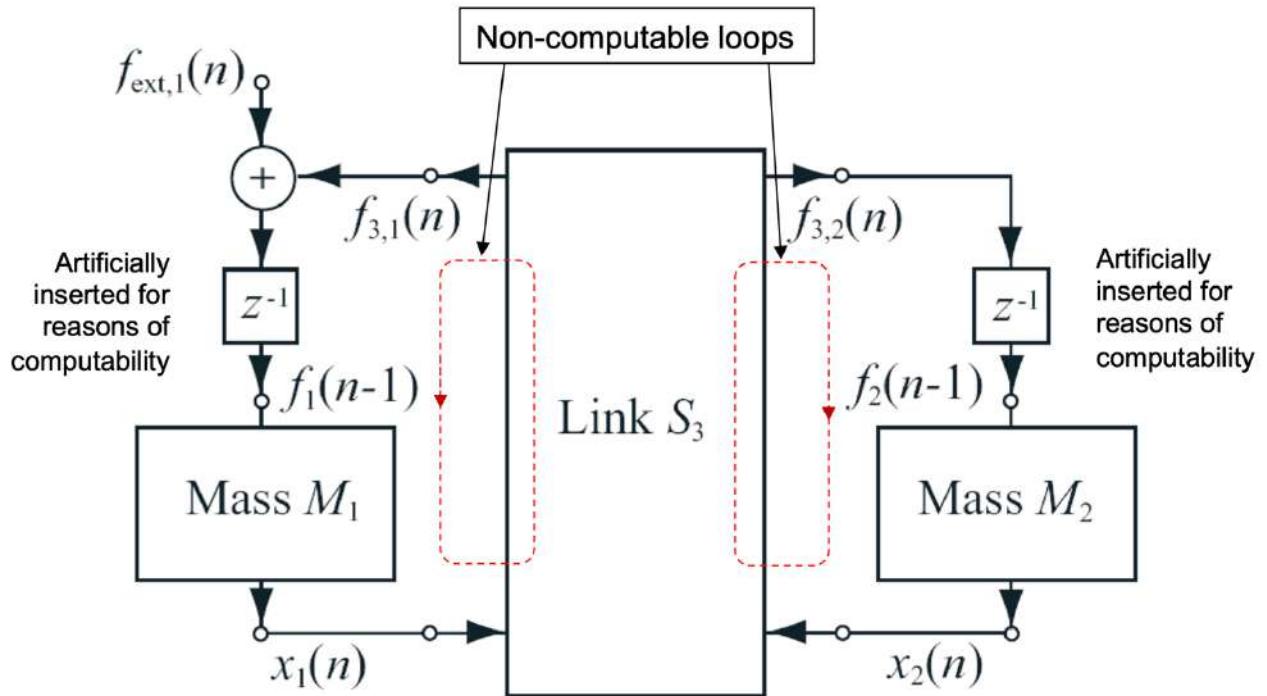


One the right, the block representing the mass. Now, we have to connect the two blocks.



Poranno! The signal  $x(n)$  depends on its value at the instant  $n$ . Obviously, it is impossible to evaluate! There is a recursively problem. We have to find a way to make the signal independent from itself. In order to address this, we insert an artificial delay.

Now the variables depend only on their previous values.



## Modal synthesis

In today's lecture we will see how to synthetise a soundwave with the modal approach.

The sound of a resonating object is represented as a linear combination of the outputs of  $N$  second-order oscillators, each representing one mode of oscillation of the object excited by a driving force or acoustic pressure. As we saw in the previous lessons, it can be regarded as a lumped physical modelling approach. It can also be interpreted with a source-filter approach (the source is the driving signal, and the filter is a bank of second-order resonators). Understanding the mathematical and physical basis of modal theory is a bit less straightforward, though it is fundamentally the same for discrete systems (networks of masses, springs, and dampers) and continuous systems (PDE in space and time). The power of modal synthesis is that it is a very general technique that can be applied to a large class of sounding physical systems as opposed to wave guide techniques, which are only suitable for the D'Alembert equation, and variations thereof.

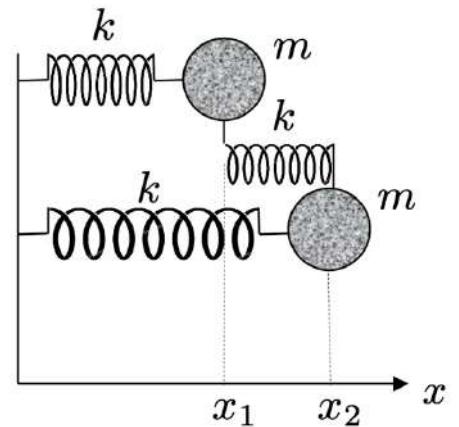
Neither modal nor WGN methods address the central issue of NL interaction btw blocks!

Let us recall the example we saw in during past lessons.

$$\begin{aligned} m\ddot{x}_1(t) + kx_1(t) + k(x_1(t) - x_2(t)) &= 0 \\ m\ddot{x}_2(t) + kx_2(t) + k(x_2(t) - x_1(t)) &= 0 \end{aligned}$$

$$\underbrace{\begin{pmatrix} m & 0 \\ 0 & m \end{pmatrix}}_{\underline{\underline{M}}} \underbrace{\begin{pmatrix} \ddot{x}_1 \\ \ddot{x}_2 \end{pmatrix}}_{\underline{\dot{y}}} + \underbrace{\begin{pmatrix} 2k & -k \\ -k & 2k \end{pmatrix}}_{\underline{\underline{K}}} \underbrace{\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}}_{\underline{y}} = \underline{\underline{0}}$$

$$\underline{\underline{M}} \underline{\dot{y}} + \underline{\underline{K}} \underline{y} = \underline{\underline{0}}$$



More generally, a network of  $N$  linear undamped oscillators can be described by the set of equations

$$\underline{\underline{M}} \underline{\dot{y}} + \underline{\underline{K}} \underline{y} = \underline{f}_{ext}$$

Where  $\underline{\underline{M}}$  is the mass matrix (generally diagonal) and  $\underline{\underline{K}}$  the stiffness matrix (generally non-diagonal as points are coupled through strings).

Let us consider the homogenous equation  $\underline{f}_{ext} = \underline{\underline{0}}$  and look for a factorised solution of the form

$$\underline{y}(t) = \underline{s} \cdot \sin(\omega t + \phi)$$

By substituting it into the  $\underline{\underline{M}} \underline{\dot{y}} + \underline{\underline{K}} \underline{y} = \underline{\underline{0}}$  equation, we get the generalised eigenvalue problem for the matrix  $\underline{\underline{K}}$ .

$$\underline{\underline{K}} \underline{s} = \omega^2 \underline{\underline{M}} \underline{s} \rightarrow \underline{\underline{M}}^{-1} \underline{\underline{K}} \underline{s} = \omega^2 \underline{s}$$

In general, we find  $N$  pairs of eigenvalue-eigenvector  $(\omega, \underline{s})$  where:

- $\omega^2$  is an eigenvalue of  $\underline{\underline{M}}^{-1} \underline{\underline{K}}$
- $\underline{s}$  is the associated eigenvector

- $\omega^2$  eigenvalues are orthogonal with respect to the mass and the stiffness matrices

$$\underline{s}_i^T \underline{\underline{M}} \underline{s}_j = \delta_{i,j} m_i \quad \text{and} \quad \underline{s}_i^T \underline{\underline{K}} \underline{s}_j = \delta_{i,j} k_i, \quad k_i = \omega_i^2 m_i$$

Here  $m_i$  is the modal stiffness and  $k_i$  the modal mass

- The  $s_i$  can now be used to define a modal transformation (a change of spatial coordinates) that turns the above system into a set of  $N$  uncoupled oscillators

$$\underline{y} = \underline{\underline{S}} \underline{q} \quad \Leftrightarrow \quad \underline{q} = \underline{\underline{S}}^{-1} \underline{y} = \underline{\underline{S}}^T \underline{y}$$

where

$$\underline{\underline{S}} = [s_1 | \dots | s_N] \text{ is orthogonal} \Rightarrow \underline{\underline{S}} \underline{\underline{S}}^T = \underline{\underline{I}}$$

Let us consider now the external force. By pre-multiplying and post-multiplying  $\underline{\underline{M}}$  and  $\underline{\underline{K}}$  to  $\underline{\underline{S}}$ . In this way, we change their system of coordinates (the bases) and we make the result of the product a diagonal matrix.

$$\begin{aligned} \underline{\underline{M}} \ddot{\underline{y}} + \underline{\underline{K}} \underline{y} = \underline{f}_{ext} &\xrightarrow{\underline{y} = \underline{\underline{S}} \underline{q}} \underbrace{\underline{\underline{M}} \underline{\underline{S}} \ddot{\underline{q}} + \underline{\underline{K}} \underline{\underline{S}} \underline{q}}_{\underline{\underline{M}}_q \ddot{\underline{q}} + \underline{\underline{K}}_q \underline{q}} = \underline{f}_{ext} \Rightarrow \underbrace{\underline{\underline{S}}^T \underline{\underline{M}} \underline{\underline{S}} \ddot{\underline{q}}}_{\underline{\underline{M}}_q \ddot{\underline{q}}} + \underbrace{\underline{\underline{S}}^T \underline{\underline{K}} \underline{\underline{S}} \underline{q}}_{\underline{\underline{K}}_q \underline{q}} = \underline{\underline{S}}^T \underline{f}_{ext} \\ &\Rightarrow \underline{\underline{M}}_q \ddot{\underline{q}} + \underline{\underline{K}}_q \underline{q} = \underline{\underline{S}}^T \underline{f}_{ext} \end{aligned}$$

We have obtained a system of uncoupled oscillators with frequencies  $\omega_i$  (decomposed in the eigenfrequencies of the system).  $m_i$  and  $k_i$  represent the masses and the stiffness of these modes.

This is what happens in the string equation with a force that acts on the string.

$$\mu \frac{\partial^2 y}{\partial t^2}(x, t) - T \frac{\partial^2 y}{\partial x^2}(x, t) = f_{ext}(x, t)$$

We assume that a normal mode is a factorised solution  $y(x, t) = s(x)q(t)$  called standing wave. Let us now substitute the solution into the equation. We multiply everything by  $s_n(x)$  and we integrate over  $x$ .

$$\begin{aligned} \left[ \mu \int_0^L s_n^2(x) dx \right] \ddot{q}_n(t) - \left[ T \int_0^L s_n''(x) s_n(x) dx \right] q_n(t) &= \int_0^L s_n(x) f_{ext}(x, t) dx \\ \left[ \mu \int_0^L s_n^2(x) dx \right] \ddot{q}_n(t) - T \left[ \begin{array}{l} \underbrace{s_n'(x)s_n(x)|_0^L}_{\substack{\text{Identically zero for} \\ \text{fixed (or even free)}}} - \int_0^L [s_n'(x)]^2 dx \end{array} \right] q_n(t) &= \int_0^L s_n(x) f_{ext}(x, t) dx \end{aligned}$$

The equation for the  $n^{th}$  mode is that of a 2<sup>nd</sup>-order oscillator with

$$m_n = \mu \int_0^L s_n^2(x) dx \quad k_n = T \int_0^L [s'_n(x)]^2 dx$$

For an ideal string, it becomes

$$s_n(x) = \sin\left(\frac{n\pi x}{L}\right) \quad m_n = \frac{\mu L}{2} \quad k_n = \frac{TL}{2}$$

The equation for the  $n$ th mode is therefore that of a 2<sup>nd</sup>-order oscillator with

$$m_n \ddot{q}_n(t) + k_n q_n(t) = F_{ext}(t)$$

where:

- $m_n = \mu \int_0^L s_n^2(x) dx$
- $k_n = T \int_0^L [s'_n(x)]^2 dx$
- $F_{ext}(t) = \int_0^L s_n(x) f_{ext}(x, t) dx$

For the ideal string the modal shapes are simply

$$s_n(x) = \sin\left(\frac{n\pi x}{L}\right) \quad m_n \underset{\substack{\equiv \\ \text{density does not change} \\ \text{from point to point}}}{=} \frac{\mu L}{2} \quad k_n \underset{\substack{\equiv \\ \text{tension does not change} \\ \text{from point to point}}}{=} \frac{T\boxed{\square}}{2}$$

The shape also defines how a driving force acts on the mode. For example, let us consider an input force as a Dirac delta concentrated in a specific point  $x_{in}$  of the string and on a function that depends on time

$$f_{ext}(x, t) = \delta(x - x_{in}) u(t)$$

Let us introduce this in the expression of  $F_{ext}(t)$

$$F_{ext}(t) = \int_0^L s_n(x) f_{ext}(x, t) dx = \int_0^L s_n(x) \delta(x - x_{in}) u(t) dx = s(x_{in}) u(t)$$

If  $x_{in}$  is a node of the mode, then no force is transmitted to it.

We can also try to create a virtual pick-up, so an object that measures the vibration of the string at a specific point. We can do that, by sampling the input wave shape at the point  $x_{out}$  and multiplying this wave shape by  $q_n(t)$  and sum all the modes together. The interesting point on a string, that can be sampled, is the one close to the bridge, where the vibration is transferred to the resonator.

## DT mechanical oscillators

Until now we have not considered the presence of dissipation and dispersion. They will be proportional to the first order derivative.

$$\ddot{q}(t) + 2\alpha\dot{q}(t) + \omega_0^2 q(t) = \frac{1}{m} f_{mode}(t)$$

- With frequency  $\omega_0 = \frac{k}{m}$
- And loss factor  $\alpha = \frac{r}{m}$

As we can notice, both of them depends on the geometry and material of the object.

In the Laplace domain it becomes:

$$Q(s) = H(s)F_{mode}s(s) \quad \text{where } H(s) = \frac{m^1}{s^2 + 2\alpha s + \omega_0^2}$$

The force  $f_{mode}$  that is “felt” by a single mode depends on the modal shape and on the spatial force distribution and it is scaled by the modal mass  $m$ . On the other hand, the displacement  $y(x, t)$  is a linear combination of the modes  $q(t)$ , where the weights are the modal shapes  $s(x)$ .

Anyway, in order to construct a modal synthesiser, we first need to construct a DT-equivalent of the 2<sup>nd</sup>-order oscillator. We can discretise the ODE with the numerical methods examined before.

- Impulse invariant method:

$$H(z) = \frac{\left[ T_s \left( \frac{e^{-\alpha T_s}}{m \omega_r} \right) \sin(\omega_r T_s) \right] z^{-1}}{1 - [2e^{-\alpha T_s} \cos(\omega_r T_s)] z^{-1} + e^{-2\alpha T_s} z^{-2}}$$

- Backward Euler method:

$$H(z) = \frac{\frac{1}{m(F_s^2 + 2\alpha F_s + \omega_0^2)}}{1 - \frac{2F_s(\alpha + F_s)}{F_s^2 + 2\alpha F_s + \omega_0^2} z^{-1} + \frac{F_s^2}{F_s^2 + 2\alpha F_s + \omega_0^2} z^{-2}}$$

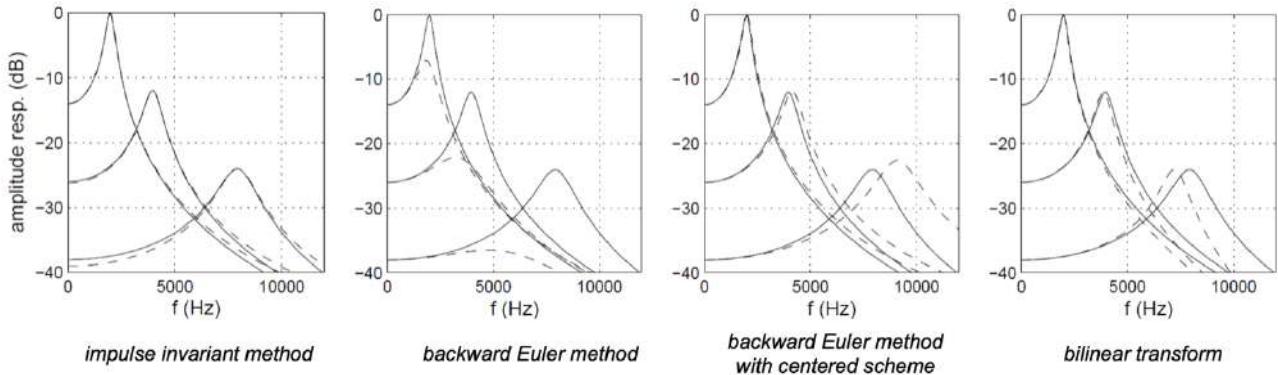
- Backward Euler method with centred scheme:

$$H(z) = \frac{\frac{T_s^2}{m} z^{-1}}{1 + [\omega_0^2 T_s^2 + 2\alpha T_s - 2] z^{-1} + 1[1 - 2\alpha T_s] z^{-2}}$$

- Bilinear transform:

$$H(z) = \frac{\left[ \frac{1}{m(4F_s^2 + 4\alpha F_s + \omega_0^2)} \right] (1 + 2z^{-1} + z^{-2})}{1 + \frac{2(\omega_0^2 - 4F_s^2)}{4F_s^2 + 4\alpha F_s + \omega_0^2} z^{-1} + \frac{4F_s^2 - 4\alpha F_s + \omega_0^2}{4F_s^2 + 4\alpha F_s + \omega_0^2} z^{-2}}$$

These are the amplitude responses of a 2<sup>nd</sup> order oscillator with constant mass and quality factor, and  $\omega_0 = 2, 4, 8 \text{ kHz}$ : continuous-time responses (solid lines) and discrete-time responses (dashed lines)



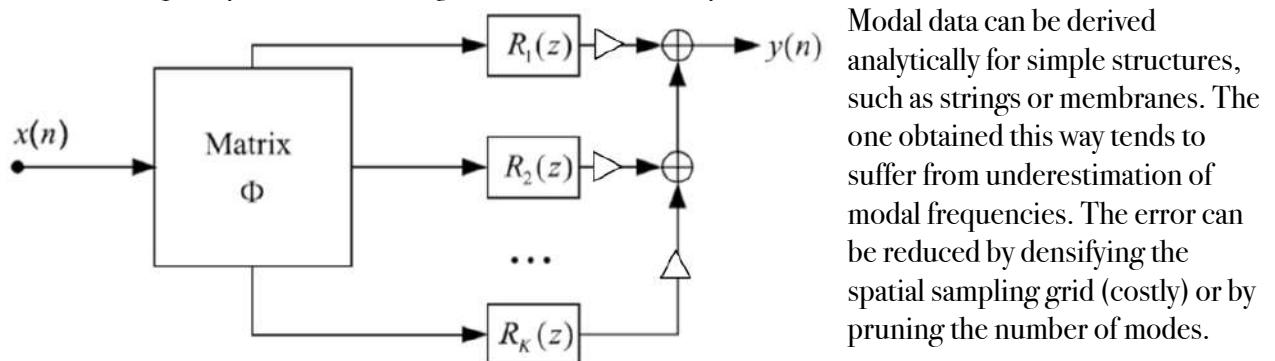
## Modal analysis

But how do we determine the modal parameters? For DT system of N point masses with linear interaction forces, modal parameters are found in closed form through standard matrix calculation. Most systems of interest do not fit these assumptions. For some distributed systems, particularly for symmetrical problems with simple boundary conditions, the PDE describing the system can be solved analytically, giving the modal parameters. Alternatively, either accurate numerical simulations (e.g. wave-guide mesh methods) or “real” physical measurements can be used.

When analytical methods cannot be applied, modal parameters can be estimated by extracting modal data from a recorded audio signal (by estimating resonances (centre frequency and quality factor) from a given signal (use LPC, peak tracking, etc.) or by simulating the response of an object with FDM (finite difference methods) or FEM (finite elements methods). In this last case a spatial discretisation has to be performed therefore only a finite number of modes can be estimated. Modal data obtained in this way suffers from underestimation of modal frequencies.

## Modal synthesis

For what it concerns the synthesis, we can interpretate the input  $x(n)$  as a force. The output  $y(n)$  will be the velocity caused by the input force that is observed at one point. In the middle, there is a series of blocks that “transforms” the force into a velocity. Let us discuss the model more deeply: following the input, we have a Matrix  $\phi$  that maps the force onto a parallel resonator bank. Each resonator represents a single mode. Resonance frequency, bandwidth and gain define the modal synthesis model.



## From Digital WaveGuides to WaveGuide Networks

The string

Let us see how to move from a digital WaveGuides to a WaveGuide network starting from an example: the equation of an ideal string. We saw many (ma many many eh) times the demonstration, so we skip it going directly to the result.

$$\frac{\partial^2 y}{\partial x^2}(x, t) = \frac{1}{c^2} \frac{\partial^2 y}{\partial t^2}(x, t) \quad c = \sqrt{\frac{T}{\mu}}$$

If we plug in either of the two travelling components, it satisfies the wave equation

$$y(x, t) = y_r(ct - x)$$

$$y(x, t) = y_l(ct + x)$$

$$\frac{\partial y_r}{\partial x}(ct - x) = -\frac{1}{c} \frac{\partial y_r}{\partial t}(ct - x)$$

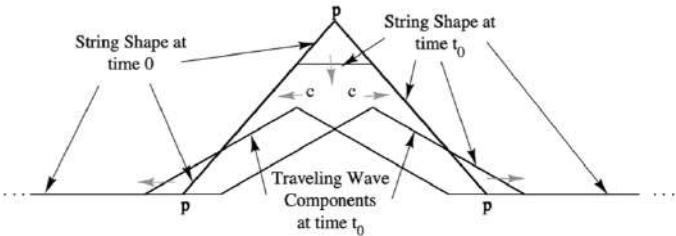
$$\frac{\partial y_l}{\partial x}(ct + x) = \frac{1}{c} \frac{\partial y_l}{\partial t}(ct + x)$$

$$\frac{\partial^2 y_r}{\partial x^2}(ct - x) = \frac{1}{c^2} \frac{\partial^2 y_r}{\partial t^2}(ct - x)$$

$$\frac{\partial^2 y_l}{\partial x^2}(ct + x) = \frac{1}{c^2} \frac{\partial^2 y_l}{\partial t^2}(ct + x)$$

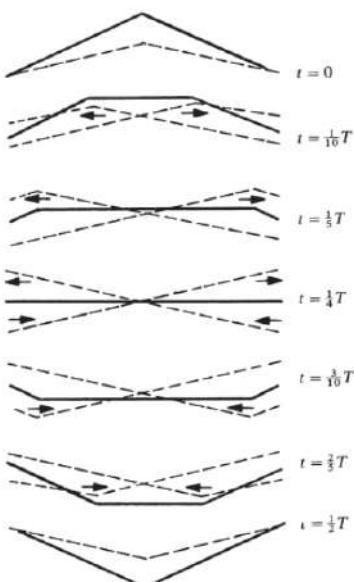
Let us see what happens to an infinitely long string plucked simultaneously at three points marked  $p$ .

As we can see in the picture, the initial displacement can be seen as the sum of two identical triangular pulses. At time  $t_0$  the travelling waves centres will be separated by  $2ct_0$  meters. The string is not moving where the travelling waves overlap at same slope.



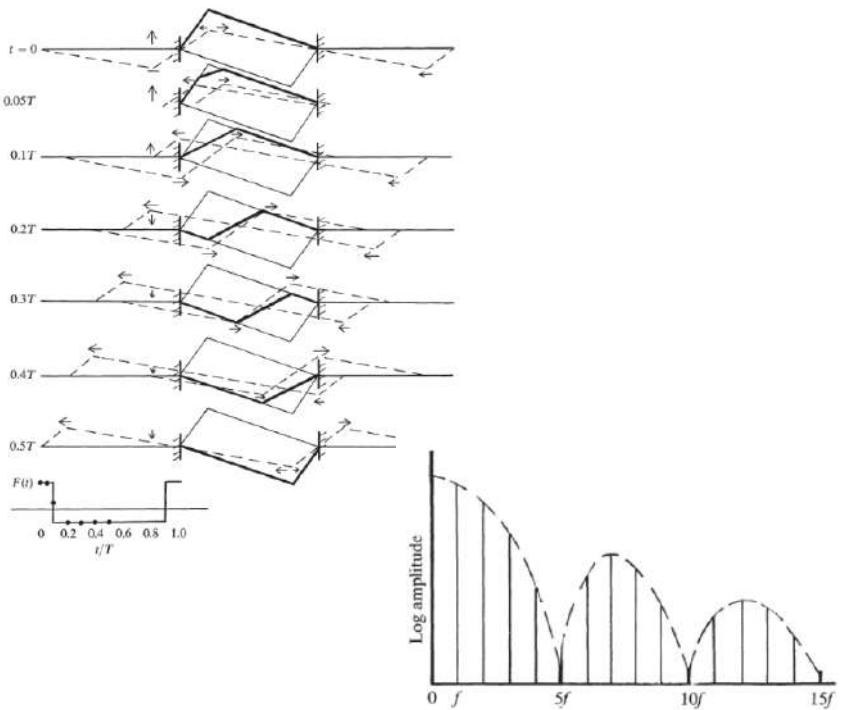
Below, we report two different type of motions.

String plucked at its midpoint through one half cycle.



Motion can be thought of as due to two pulses travelling in opposite directions.

Motion of a string plucked one-fifth of the distance from one end and its relates spectrum,



Instead of sampling two variables (space and time), we only have one variable to sample. In fact, we do not sample the wave equation (FD), we sample its general solution. The temporal sampling interval is  $T_s$  [s] from which the temporal sampling rate will be  $f_s = \frac{1}{T_s}$  [Hz]. The spatial sampling rate is  $X_s = cT_s$  [ $\frac{m}{s}$ ], systolic grid. For a vibrating string with length  $L$  and fundamental frequency  $f_0$  we have

$$c = f_0 \cdot 2L \quad \left( \frac{\text{periods}}{\text{sec}} \cdot \frac{\text{meters}}{\text{period}} = \frac{\text{meters}}{\text{sec}} \right)$$

$$\text{so that } X_s = cT_s = 2Lf_0 \frac{1}{f_s} = \frac{2Lf_0}{f_s}$$

Thus, the number of spatial samples along the string is  $\frac{L}{X_s} = \frac{f_s}{2f_0}$ . As we can notice, the number of spatial samples corresponds to the number of string harmonics.

Example:

- Spatial sampling interval for (1/2) CD-quality digital model of Les Paul electric guitar (strings  $\approx 660\text{mm}$ )
- Low E string:  $X_s = \frac{Lf_0}{\frac{f_s}{2}} = L \cdot \frac{82.4}{22050} \approx 2.5\text{mm}$
- High E string:  $X_s \approx 10\text{mm}$  (two octaves higher and same length)
- Low E string:  $\frac{\frac{f_s}{2}}{f_0} = \frac{22050}{82.4} = 268$  harmonics (spatial samples)
- High E string: 67 harmonics (spatial samples)
- Number of harmonics = number of oscillators required in additive synthesis
- Number of harmonics = number of two-pole filters required in subtractive, modal, or source-filter decomposition synthesis
- Digital waveguide model needs only one delay line (length  $2L$ )

Then, the sound is propagated in air:

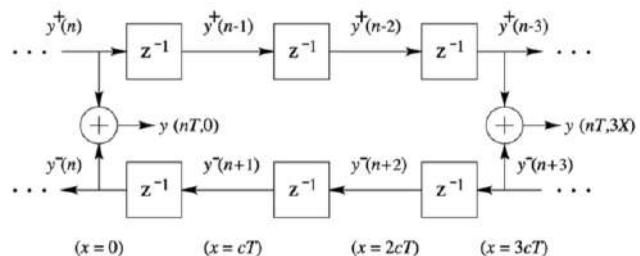
- Speed of sound  $c \approx 331 \frac{\text{m}}{\text{s}}$
- $X_s = \frac{331}{44100} = 7.5\text{ mm}$
- Spatial sampling rate:  $\frac{1}{X_s} = 133 \frac{\text{samples}}{\text{m}}$
- Sound speed in air is comparable to that of transverse waves on a guitar string (faster than some strings, slower than others)
- Sound travels much faster in most solids than in air
- Longitudinal waves in strings travel faster than transverse wave

Let us discretise it

$$\begin{aligned} x &\rightarrow x_m = mX_s \\ t \rightarrow t_n &= nT_s \Rightarrow y(x_n, t_n) = y_r(cnT_s - mX_s) + y_l(cnT_s + mX_s) = y^+(n-m) + y^-(m+m) \\ X_s &= cT_s \end{aligned}$$

Where:

- $y^+$  is the output of  $m$ -samples delay line
- $y^-$  is the output of  $m$ -sample delay line



Notice that  $y(nT_s, mX_s) = y^+(n - m) + y^-(m + m)$  implies that:

- Position  $x_m = mX_s = mcT_s$  is eliminated from the simulation
- Position  $x_m$  remains laid out from left to right
- Left- and right-going traveling waves must be summed to produce a physical output

Let us now highlight an important point: DT simulation is exact at the sampling instants, to within the numerical precision of the samples themselves.

To avoid aliasing associated with sampling:

- Require all initial waveshapes be bandlimited to  $(-\frac{f_s}{2}, \frac{f_s}{2})$
- Require all external driving signals be similarly bandlimited
- Avoid nonlinearities or keep them “weak”
- Avoid time variation or keep it slow
- Use plenty of lowpass filtering with rapid high-frequency roll-off in severely nonlinear and/or time-varying cases
- Prefer “feed-forward” over “feed-back” around nonlinearities and/or modulations when possible

Let us look now for the wave variables, we write the slope waves.

$$y'(x, t) = \frac{\partial}{\partial x} y(x, t) = y'_r(ct - x) + y'_l(ct + x) = -\frac{1}{c} \dot{y}_r(ct - x) + \frac{1}{c} \dot{y}_l(ct + x)$$

Notice that computing the derivative of something that depends on (ct-x) or (ct+x) whether the derivative is taken with respect to space or time! We are not doing differentiating the PDE, we are differentiating traveling waves (general solution of the PDE)!

$$y'(x, t) = -\frac{1}{c} \dot{y}^+(n - x) + \frac{1}{c} \dot{y}^-(n + x)$$

Where:

$$\begin{aligned} y'^+ &= -\frac{1}{c} v^+ & \Leftrightarrow & v^+ = -cy'^+ \\ y'^- &= \frac{1}{c} v^- & & v^- = cy'^- \end{aligned}$$

From the slope waves written above, we can define the new wave variables, which are in physical units of force:

$$\begin{aligned} f^+ &\triangleq -Ty'^+ & \Rightarrow & f^+ \triangleq \frac{T}{c} v^+ & \Rightarrow & f^+ \triangleq Z_0 v^+ \\ f^- &\triangleq -Ty'^- & & f^- \triangleq -\frac{T}{c} v^- & & f^- \triangleq -Z_0 v^- \end{aligned}$$

where

$$c = \sqrt{\frac{T}{\mu}} \Rightarrow \frac{T}{c} = \sqrt{T\mu} \triangleq Z_0$$

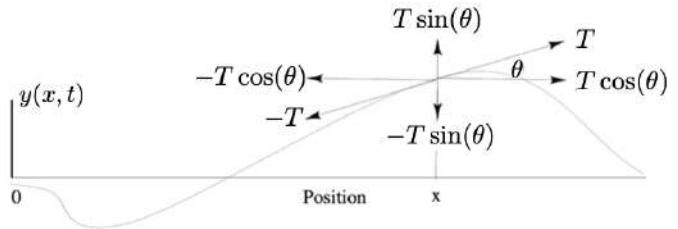
Let us see a physical interpretation

The vertical force acting on the left can be written as

$$f_l(x, t) = T \sin(\theta) \approx T \tan(\theta) = Ty'(x, t)$$

and the opposite force, acting on the right

$$f_r(x, t) = -T \sin(\theta) \approx -T \tan(\theta) = -Ty'(x, t)$$



These forces must cancel since a nonzero net force on a massless point would produce infinite acceleration. To unify vibrating strings with acoustic tubes, we choose the force acting on the right as our force wave variable (similar to longitudinal pressure in an acoustic tube)

$$f(x, t) \triangleq f_r(x, t) = -Ty'(x, t) = \frac{T}{c} [\dot{y}_r(ct - x) - \dot{y}_l(ct + x)]$$

Now, let us summarise the variables

- Velocity waves

$$\begin{aligned} v^+(n) &\triangleq \dot{y}^+(n) \\ v^-(n) &\triangleq \dot{y}^-(n) \end{aligned}$$

- Force waves

$$\begin{aligned} f^+(n) &\triangleq -Ty'^+(n) \\ f^-(n) &\triangleq -Ty'^-(n) \end{aligned}$$

- Wave impedance

$$Z_0 \triangleq \frac{f^+}{v^+} \triangleq -\frac{Ty'^+}{\dot{y}^+} = \frac{T}{c}$$

$$\text{as } y'^+ = -\frac{1}{c} \dot{y}^+, \text{ where } c = \sqrt{\frac{T}{\mu}} \rightarrow Z_0 = \sqrt{T\mu} = \frac{T}{c} = \mu c$$

The relation between the three is the Ohm's law for traveling waves

$$\begin{aligned} f^+(n) &= Z_0 v^+(n) \\ f^-(n) &= -Z_0 v^-(n) \end{aligned}$$

In the previous slides we adopted specific changes of variables from Kirchhoff pairs ( $f, v$ ) to Wave pairs (incident, reflected). This transformation is invertible (W<sub>2</sub>K and K<sub>2</sub>W mappings):

- K<sub>2</sub>W

$$\begin{pmatrix} f^+ \\ f^- \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & Z_0 \\ 1 & -Z_0 \end{pmatrix} \begin{pmatrix} f \\ v \end{pmatrix}$$

- W<sub>2</sub>K

$$\begin{pmatrix} f \\ v \end{pmatrix} = \begin{pmatrix} 1 & \frac{1}{Z_0} \\ 1 & -\frac{1}{Z_0} \end{pmatrix} \begin{pmatrix} f^+ \\ f^- \end{pmatrix}$$

## Acoustic Tube

The case of acoustic tubes is similar:

- Variables: pressure  $p$  and flow  $u$  (volume velocity)

$$\begin{aligned} p^+(ct - x) &= Z_0 u^+(ct - x) \\ p^-(ct + x) &= -Z_0 u^-(ct + x) \end{aligned}$$

$$Z_0 = \frac{\rho_{air} c}{S}$$

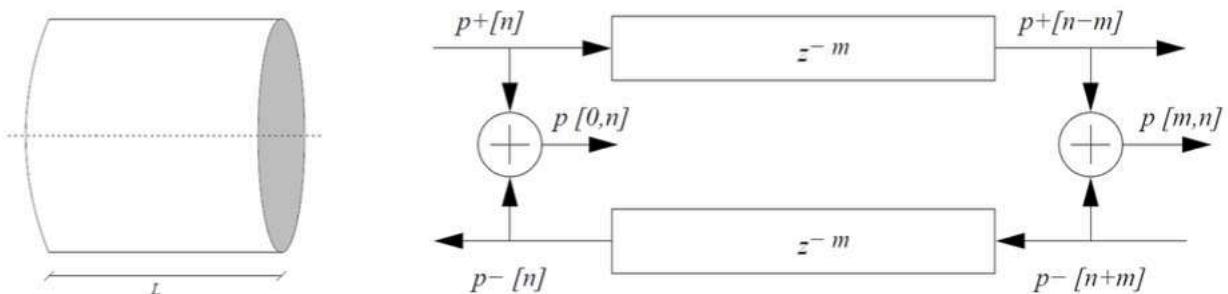
- Conical geometry

$$\begin{aligned} P^+(r, s) &= Z_0 U^+(r, s) \\ P^-(r, s) &= -Z_0 U^-(r, s) \end{aligned}$$

$$Z_0(s) = \frac{\rho_{air} c}{S} \cdot \frac{rs}{rs + c}$$

As we can notice, in this case the impedance is function of the position.

We can write the equations of the acoustic tube in terms of digital waveguides. What we obtain, by considering an ideal lossless cylindrical bore and discretising the pressure  $p$  both in time and in space, is the structure below.



The discretisation is made in the same way we saw for the acoustic string

$$x \mapsto mX_s \quad t \mapsto nT_s$$

Where:

- $X_s = cT_s$
- $L = mX_s$

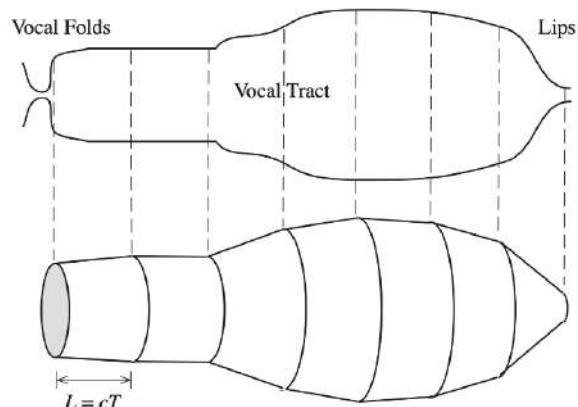
What we obtain is

$$p(mX_s, nT_s) = p_r(ncT_s - mX_s) + p_l(ncT_s + mX_s) = p_r((n - m)cT_s) + p_l((n + m)cT_s)$$

That can be written as

$$p(m, n) = p^+(n - m) + p^-(n + m)$$

It could happen that the cross section of a structure is not constant. When this happens, the tube is approximated with a series of conical bores. A typical example is the vocal tract: having an irregular structure, it is approximated with a series of conical bores. We notice that the distance between one bore to the other is always  $L = cT$ . As the structure is approximated with conical and cylindrical bores, we have to find a way to model them.



### Cylindrical bore

The general 3D form of the D'Alembert wave equation is given by  $\ddot{p} = c^2 \nabla^2 p$ . Using cylindrical coordinates one can show that for cylindrical bores 1D longitudinal pressure waves in the  $z$  direction are described by the 1D d'Alembert equation with  $z$  in place of  $x$  ( $p$ =acoustic pressure).

$$\ddot{p} = c^2 p''$$

Using spherical coordinates one can show that for conical bores 1D spherical pressure waves are described by the equation

$$\frac{1}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial P}{\partial r} \right) (r, t) = \frac{1}{c^2} \frac{\partial^2 P}{\partial t^2} (r, t)$$

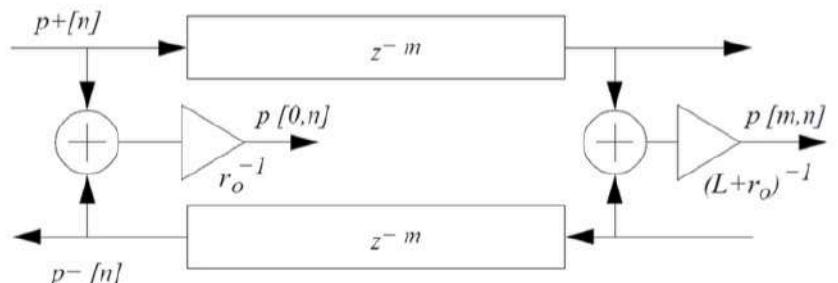
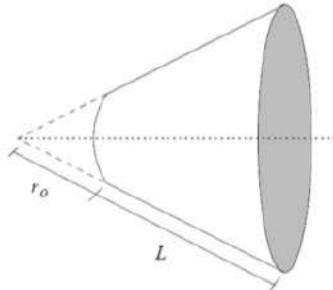
where  $P(r)$  is the acoustic pressure as a function of the radial coordinate. Using the substitution  $P = \frac{p}{r}$  we obtain again the 1D d'Alambert equation:

$$\ddot{p} = c^2 p''$$

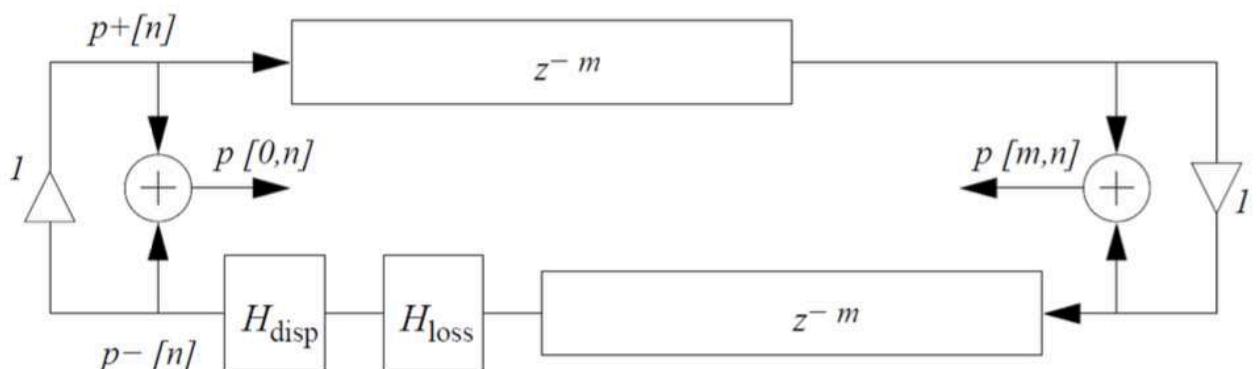
### Conical bore

The conical bore can be simply represented by

$$\ddot{p} = c^2 p''$$



This time again we have to cope with the problem of the boundary condition, tuning, dissipation and dispersion.



In this scheme we have:

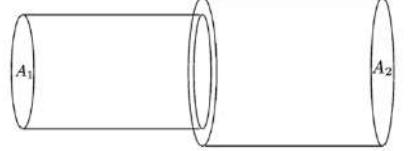
- The triangles that represent the boundary conditions
- The block  $H_{disp}$  that represents the dissipation
- The block  $H_{loss}$  that represents the losses

So far, we had considered a uniform medium. When discontinuities are encountered, the wave impedance changes, and signal scattering occurs. We need to define how to connect two WaveGuide sections.

### Kelly-Lochbaum junction

It is funny to notice that Kelly Lochbaum is a Canadian football player. Anyway, we consider the connection of two cylindrical bores of section  $A_1$  and  $A_2$  and admittances

$$\Gamma_1 = \frac{1}{Z_1} = \frac{A_1}{\rho_{air}c} \quad \Gamma_2 = \frac{1}{Z_2} = \frac{A_2}{\rho_{air}c}$$



We need to set physical constraints on the Kirchhoff variables  $p$  and  $u$  at the junction. The Kirchhoff laws are given by the continuity conditions. Then we have

$$u_1 + u_2 = 0 \quad p_1 = p_2 = p_J$$

In order to satisfy the continuity conditions, the discontinuity in the medium (change in characteristic impedance), will cause wave scattering. Traveling waves will partially bounce back and partially travel through on either side of the discontinuity. We want to find the input/output relationships that describe the junction that implements scattering in the WD domain. We do so by replacing the definition of waves

$$\begin{aligned} p_1 &= p_1^+ + p_1^- \\ p_2 &= p_2^+ + p_2^- \end{aligned} \quad \begin{aligned} u_1 &= \Gamma_1(p_1^+ - p_1^-) \\ p_2 &= \Gamma_2(p_2^+ - p_2^-) \end{aligned}$$

Into the continuous equations and we express the outputs ( $p_1^-$  and  $p_2^-$ ) in function of the inputs ( $p_1^+$  and  $p_2^+$ ). One way to compute the scattering relations is the following:

1. We start from the continuity equations (low and pressure preservation)

$$u_1 + u_2 = 0 \quad p_1 = p_2 = p_J$$

2. We replace the definition of pressure waves into the first continuity equation (nodes) we get

$$0 = (u_1^+ + u_1^-) + (u_2^+ + u_2^-) = \Gamma_1(p_1^+ - p_1^-) + \Gamma_2(p_2^+ - p_2^-) = \Gamma_1(2p_1^+ - p_J) + \Gamma_2(2p_2^+ - p_J)$$

3. From this we can write the junction pressure as a function of the incoming pressure waves  $p_1^+$   $p_2^+$

$$p_J = 2 \frac{\Gamma_1 p_1^+ + \Gamma_2 p_2^+}{\Gamma_1 + \Gamma_2}$$

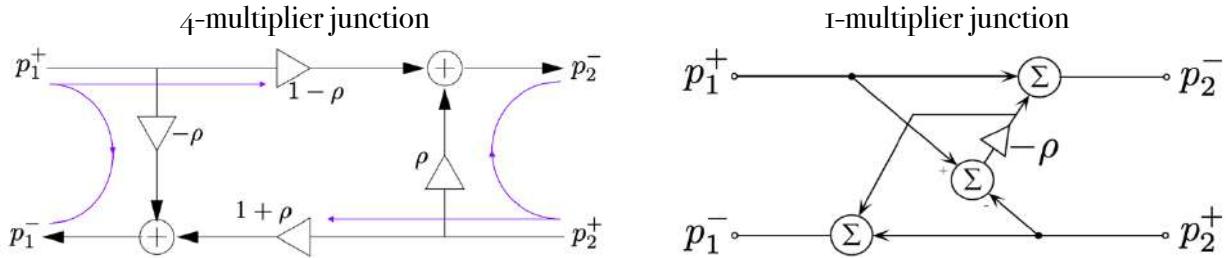
4. Using the expression of the Junction Pressure we can compute the outgoing pressure waves as

$$p_1^- = p_J - p_1^+ = -\frac{\Gamma_2 - \Gamma_1}{\Gamma_2 + \Gamma_1} p_1^+ + \frac{2\Gamma_2}{\Gamma_2 + \Gamma_1} p_2^+ \quad p_2^- = p_J - p_2^+ = +\frac{2\Gamma_1}{\Gamma_2 + \Gamma_1} p_1^+ + \frac{\Gamma_2 - \Gamma_1}{\Gamma_2 + \Gamma_1} p_2^+$$

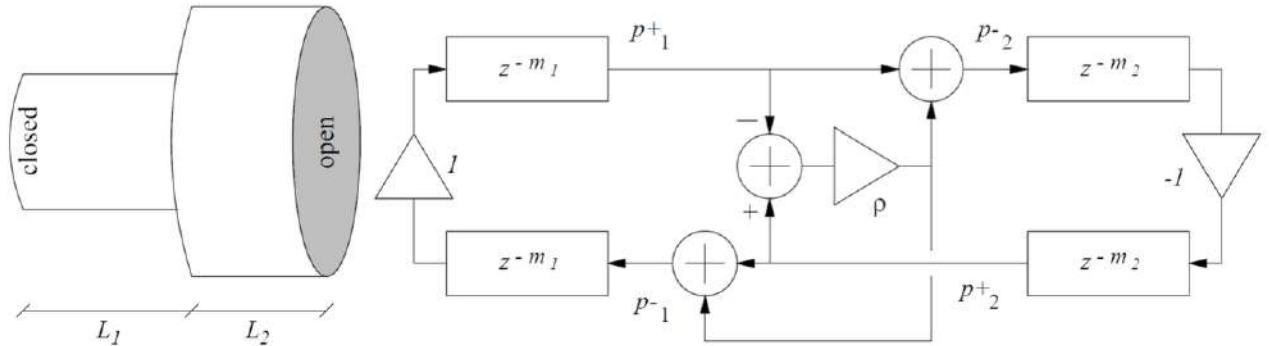
5. And, finally, we can write the scattering equations (outputs as a function of the inputs) as follows:

$$\begin{aligned} p_1^- &= -\rho p_1^+ + (1 + \rho) p_2^+ \\ p_2^- &= (1 - \rho) p_1^+ + \rho p_2^+ \end{aligned} \quad \begin{aligned} \text{Reflection coefficient} \\ \rho \triangleq \frac{\Gamma_2 - \Gamma_1}{\Gamma_2 + \Gamma_1} \end{aligned}$$

This specific case can be represented in two ways



So, the overall model can be represented as it follows



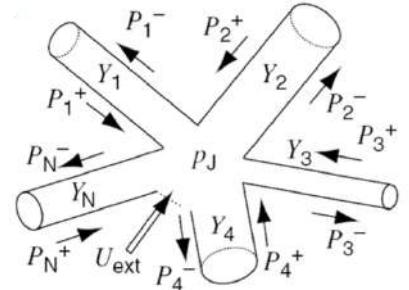
### N-dimensional and loaded junctions

The above results can be readily extended to higher dimensions: we consider a parallel junction of  $N$  acoustics bores whose relationship can be written as

$$\underline{p}^- = \underline{\underline{A}} \underline{p}^+$$

where, for KVL and KCL, being a parallel relation, we have

- Loops:  $p_1 = p_2 = \dots = p_N = p_J$
- Nodes:  $u_1 + u_2 + \dots + u_N = 0$



The matrix we will have will be

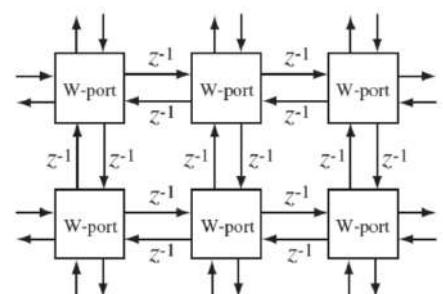
$$\underline{\underline{A}} = \begin{pmatrix} \frac{2\Gamma_1}{\Gamma_J} - 1 & \frac{2\Gamma_2}{\Gamma_J} & \dots & \frac{2\Gamma_N}{\Gamma_J} \\ \frac{2\Gamma_1}{\Gamma_J} & \frac{2\Gamma_2}{\Gamma_J} - 1 & \dots & \frac{2\Gamma_N}{\Gamma_J} \\ \vdots & \ddots & \ddots & \vdots \\ \frac{2\Gamma_1}{\Gamma_J} & \frac{2\Gamma_2}{\Gamma_J} & \dots & \frac{2\Gamma_N}{\Gamma_J} - 1 \end{pmatrix}$$

$$\Gamma_J = \sum_{l=1}^N \Gamma_l$$

We notice that, when  $N = 2$  this becomes exactly the Kelly-Lochbaum Junction.

### DWG meshes and networks

The scattering junctions enable arbitrary interconnections of delay lines and junction nodes. Stability can be easily guaranteed by keeping all elements passive (real part of each admittance positive at all frequencies). The digital waveguide mesh is a special case of DWN with a regular structure. It will constitute the base for the next lesson in which we will learn how to implement it.



**Exam:**

- Since we expect to have a mid-term exam that has not been done, the exam of 12<sup>th</sup> June will be a simplified version of the exam.
- What should I study for the exam? What is the idea behind the technique. We have to be able to describe the procedure. If we know how to derive the final result, it is a plus.
- Duration: 45 minutes for each modules.
- 1 numerical exercise and theoretical questions.

**Wave Digital Structure**

A physical model is generally specified by a set of «constitutive equations» (ODEs and/or PDEs), each describing a functional element and involving a set of «continuity constraints» that specify how such elements interact with each other. The first ones can be classified as extensive variables (pressure, voltage, force) or intensive variables (flow, current, velocity). The second ones are simply the physical laws that “connect” the variables together (Kirchhoff laws, Laws of dynamics, conservation laws in fluid-dynamics, ...). Physical models, in fact, can usually be seen as a collection of functional blocks that interact with each other through specific contact points. These blocks can be thought of as «black boxes» (even if they are inherently modelled in a distributed-parameter fashion) that we can interact with through specific interaction points. When they are modelled as PDEs (partial differential equations), we can “lumpify” them. Once we have everything “lumped” we connect the black boxes knowing that the interaction points correspond to ports of the equivalent electric circuit.

In the analogical world, this is represented by a set of equations, that needs solving. Software of circuit simulation SW such as SPICE are specialized in solving such sets of equations. The question we ask ourselves is: can we turn one such set of equation into a computable signal flow?

In order to solve this problem, we get inspiration from DWG (digital wave guide, I cannot understand why we have to talk as if we were reading a tax code) theory.

The fact that the general solution of the 1D d'Alembert equation is a pair of waves that are traveling undisturbed in opposite directions suggests some sort of «signal flow».

$$\ddot{v}(x, t) = c^2 v''(x, t) \rightarrow v(x, t) = v_r(ct - x) + v_l(ct + x)$$

Instead of using the electronical model (tension and current) we use the progressive and regressive waves that turns out to be the result of a linear and invertible mapping from the corresponding Kirchhoff (extensive-intensive) pair.

$$\begin{pmatrix} v_r \\ v_l \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & Z_0 \\ 1 & -Z_0 \end{pmatrix} \begin{pmatrix} v \\ i \end{pmatrix} \Leftrightarrow \begin{pmatrix} v \\ i \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{Z_0} & -\frac{1}{Z_0} \end{pmatrix} \begin{pmatrix} v_r \\ v_l \end{pmatrix}$$

The linear mapping depends on the characteristic impedance  $Z_0$  of the physical structure that the PDE is modelling. Then we discretise the signal transforming  $v_r$  in  $v^+$  and  $v_l$  in  $v^-$ . That is the starting point of the digital wave guides.

We also know that to connect different blocks we can use junctions. Connected to each junction there is a scattering matrix that describes the properties of the junction itself.

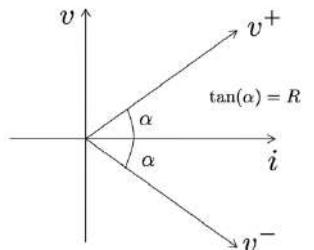
## Wave Digital Filters principles

Wave Digital Filters (WDFs) are the “lumped” counterpart of DWGs. Originally devised to model a digital filter after an analog one, they actually offer an interesting approach to physical modelling, which is valid for any port-by-port interconnections of physical blocks. Let us see how to implement them.

We make a change of variables switching from current and tension to incident and reflected waves.

$$\begin{pmatrix} v_r \\ v_l \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & R \\ 1 & -R \end{pmatrix} \begin{pmatrix} v \\ i \end{pmatrix}$$

Where “+” and “-” denote “incident” and “reflected” waves, respectively, and  $R$  is a free parameter called «reference resistance». We choose  $R$  in a way that things are simplified. We highlight that this is not an orthogonal transformation. Mathematically what we get are two arrows that are two bisectors of the cartesian plane generated by the “starting” vector. In this context,  $R$  can be seen as the tangent of the angle  $\alpha$  between the base and the vector.

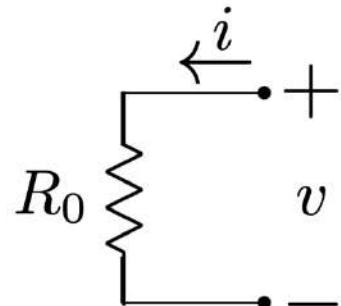


Let us see now how  $R$ , the degree of freedom changes for different elements:

### Resistor

The resistor is an element described by a simple algebraic constitutive equation

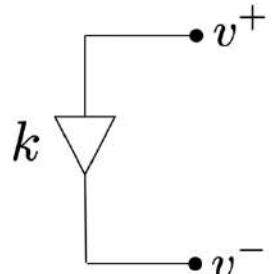
$$v = R_0 i$$



Its representation in the wave domain is obtained by simply applying the K2W mappings to the constitutive equation

$$v = R_0 i \rightarrow v^+ - v^- = R_0 \frac{v^+ - v^-}{R} \rightarrow v^- = k v^+$$

With  $k = \frac{R_0 - R}{R_0 + R}$ . We notice that the expression of  $k$  recalls the reflection coefficients.



We obtain a reflection coefficient that becomes zero when the reference resistance matches the resistor’s impedance.

$$R = R_0 \rightarrow k = 0$$

Pay attention not to make confusion between  $R$  (the degree of freedom) and  $R_0$  the value of the resistor.

### Generic impedance

From this discussion on the resistor is easy to generalise for a generic impedance. By applying the mapping, we get

$$K(s) = \frac{Z(s) - R}{Z(s) + R}$$

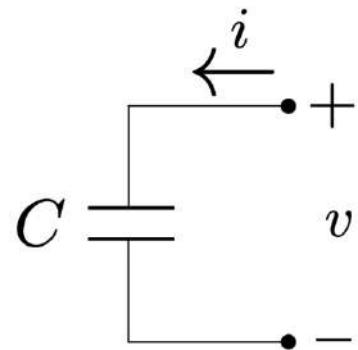
We obtain a reflection filter that can be readily discretised to  $K_d(z)$ . Assuming that  $K_d(z)$  is causal and  $k_d(n)$  the corresponding impulse response, we can easily eliminate the instantaneous input-output dependency of the reflection filter ( $k_d(0) = 0$ ) by setting  $R = z_d(0)$ , where  $z_d(n)$  is the impulse response associated to  $Z_d(s)$ , discretisation of  $Z(s)$ .

### Capacitor

We have simply to apply the capacitor Laplace equation to the case of a general impedance.

$$Z(s) = \frac{1}{sC}$$

$$K(s) = \frac{Z(s) - R}{Z(s) + R} = \frac{\frac{1}{sC} - R}{\frac{1}{sC} + R}$$

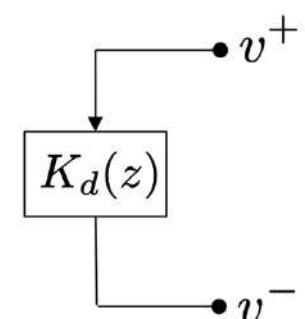


Let us now discretising it by imposing

$$s = \frac{2}{T_s} \frac{1 - z^{-1}}{1 + z^{-1}}$$

At the end we get

$$K_d(z) = \frac{p + z^{-1}}{1 + pz^{-1}} \quad \text{with } p = \frac{T - 2RC}{T + 2RC} = \frac{T - 2\tau}{T + 2\tau}$$



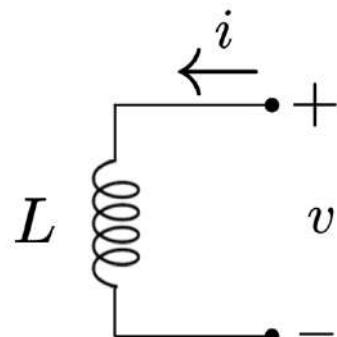
We notice that is the response of an all-pass filter! Furthermore, if  $p$  is equal to zero, we adopted a representation for which the capacitor becomes just a delay sample. At the end the reflection coefficient is nothing but a delay element in our system. So, if we impose  $R = \frac{T}{2C}$  we are done as  $K_d(z) = z^{-1}$  and the recursive dependence (instantaneous relation between  $v^-$  and  $v^+$ ) is eliminated.

### Inductor

Here again, we have simply to apply the inductor Laplace equation to the case of a general impedance.

$$Z(s) = sL$$

$$K(s) = \frac{Z(s) - R}{Z(s) + R} = \frac{sL - R}{sL + R}$$

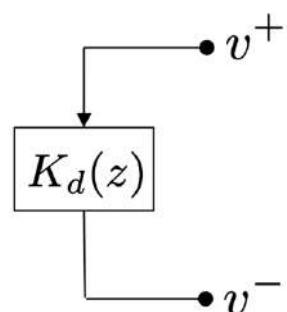


Let us now discretising it by imposing

$$s = \frac{2}{T_s} \frac{1 - z^{-1}}{1 + z^{-1}}$$

At the end we get

$$K_d(z) = \frac{p - z^{-1}}{1 - pz^{-1}} \quad \text{with } p = \frac{\frac{2L}{R} - T}{\frac{2L}{R} + T} = \frac{2\tau - T}{2\tau + T}$$



We have to avoid an instantons relation between  $v^-$  and  $v^+$ , we have to set  $p = 0$  once again. In order to achieve this result

$$\frac{2L}{R} - T = 0 \rightarrow R = \frac{2L}{T} \rightarrow K_d(z) = -z^{-1}$$

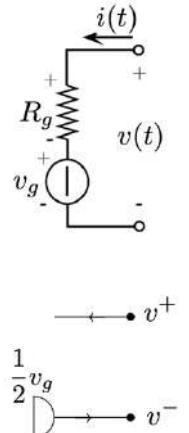
We notice that the regressive wave is nothing but the regressive wave to which we have inverted the sign.

### Voltage Generator

With the same method, we find the expression of a voltage real (with some losses) generator. What we get is

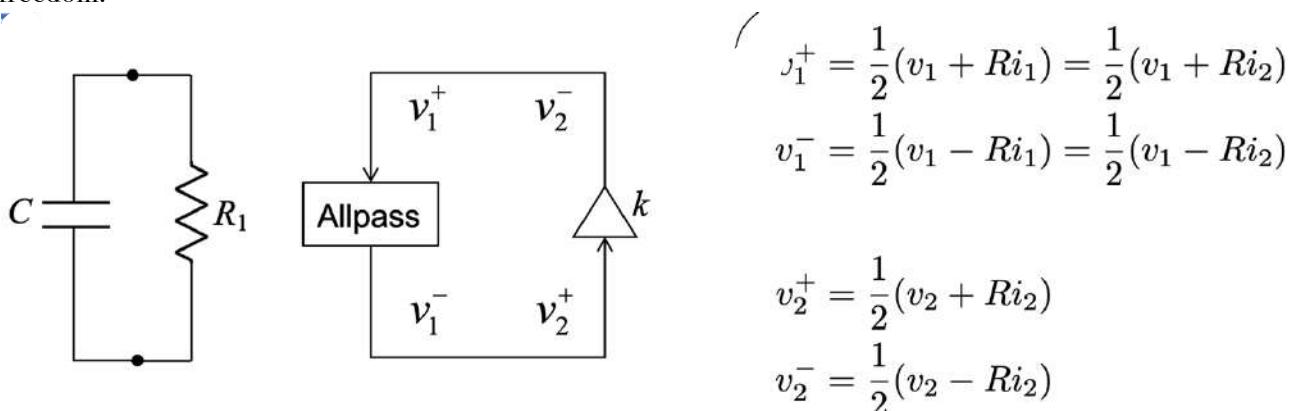
$$v^+ = \frac{1}{2} v_g + \frac{1}{2} (R_g + R)i = \frac{1}{2} v_g + R_g i$$

$$v^- = \frac{1}{2} v_g + \frac{1}{2} (R_g - R)i = \frac{1}{2} v_g$$



### Connecting two dipoles

We have to see now how to connect the dipoles. This is simply done by choosing properly the degree of freedom.



Case 1)  $R = \frac{T}{2C} \Rightarrow p = 0, k = \frac{R_1 - \frac{T}{2C}}{R_1 + \frac{T}{2C}}$

$$v_2^- = v_1^+, \quad v_1^- = v_2^+, \quad i_1 = i_2$$

We have two possible values:

1.  $R = \frac{T}{2C}$  for which the pole becomes  $p = 0$  and  $k = \frac{R_1 - \frac{T}{2C}}{R_1 + \frac{T}{2C}}$

2.  $R = R_1$  for which the pole becomes  $p = \frac{R_1 - \frac{T}{2C}}{R_1 + \frac{T}{2C}}$  and  $k = 0$

In both cases, we are able to avoid the instantaneous relationship. We want to make the block reflection free (adapted). We know that when a block is reflection free, we can connect it to a port that is not reflection free. In general, blocks that exhibit an instantaneous reflection can only be connected to a reflection-free port. We have to find a way to connect them.

## Adaptors

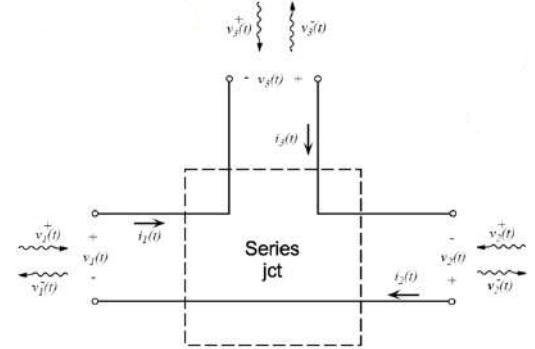
This is the problem of adaptors. We have to find a way to properly interconnects different digital waves. The solution of our problem lies in the scattering matrix. Let us see the two basic junctions:

### Series Junction

If we define two waves

$$v_k^+ = \frac{1}{2}(v_k + R_k i_k)$$

$$v_k^- = \frac{1}{2}(v_k - R_k i_k)$$



We are in the case of a series junction if the sum of the voltage drops on the port of the junction sums to zero. Below is reported the continuity equation of a series junction

$$v_1 + v_2 + v_3 = 0 \rightarrow (v_1^+ + v_1^-) + (v_2^+ + v_2^-) + (v_3^+ + v_3^-) = 0$$

$$i_1 = i_2 = i_3 \rightarrow \frac{v_1^+ - v_1^-}{R_1} = \frac{v_2^+ - v_2^-}{R_2} = \frac{v_3^+ - v_3^-}{R_3}$$

We remind once again that  $R$  is not the resistance, it is the adaptation parameters of the port. Having three parameters that allows us to perform the adaptation, we have three degrees of freedom. We have to use them, or part of them, to obtain the adaptation. If we write reflected waves (outputs) as a function of the incident waves (inputs) we obtain the scattering equation (progressive waves in function of the regressive ones).

$$\begin{pmatrix} v_1^- \\ v_2^- \\ v_3^- \end{pmatrix} = \underbrace{\left\{ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} - \begin{pmatrix} \alpha_1 & \alpha_1 & \alpha_1 \\ \alpha_2 & \alpha_2 & \alpha_2 \\ \alpha_3 & \alpha_3 & \alpha_3 \end{pmatrix} \right\}}_{\text{Scattering Matrix}} \begin{pmatrix} v_1^+ \\ v_2^+ \\ v_3^+ \end{pmatrix}$$

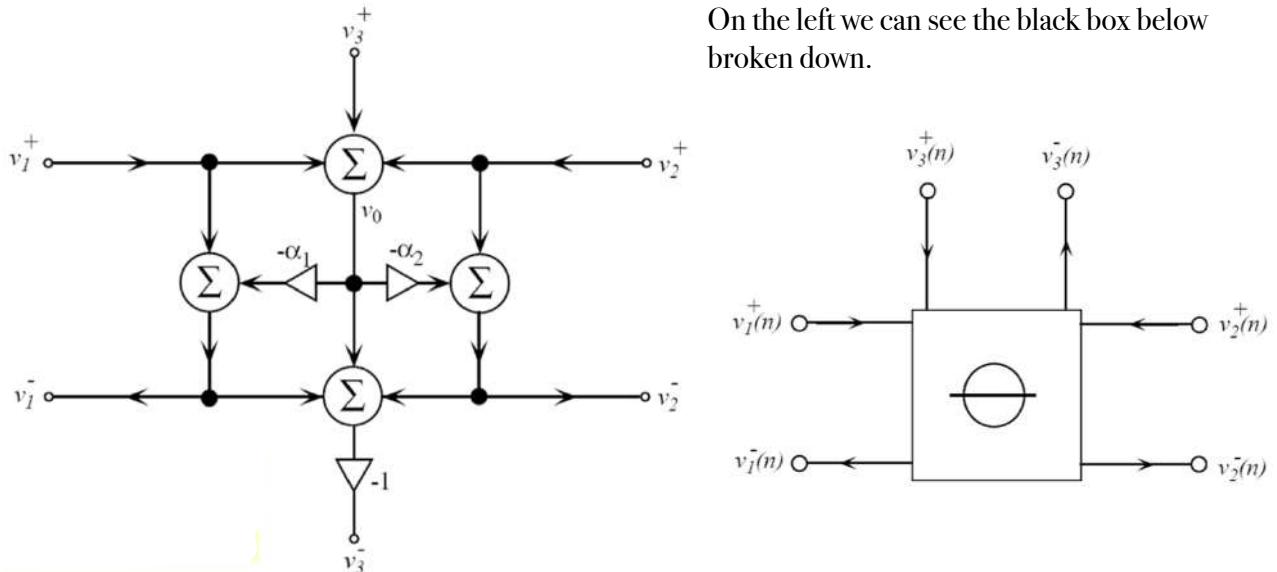
The scattering matrix is a matrix the is inherent to the topology of the system we are considering. As we can notice, we prefer to write the scattering matrix as the difference between an identity matrix and a second general matrix. Inhere,  $\alpha = \frac{2R_i}{R_1+R_2+R_3}$  is similar to the transmission coefficients. Therefore  $1 - \alpha_i$  will be the reflection coefficient. What we have is that the three ports are not independent of each other, as  $\alpha_1 + \alpha_2 + \alpha_3 = 2$ . Therefore, we need to make some derivation to see how one depends on the other.

$$\begin{cases} v_1^- = v_1^+ - \alpha_1 v_0^+ \\ v_2^- = v_2^+ - \alpha_2 v_0^+ \\ v_3^- = v_3^+ - \alpha_3 v_0^+ \\ v_0^+ = v_1^+ + v_2^+ + v_3^+ \end{cases} \rightarrow v_3 = v_3^+ - \alpha_3 v_0^+ \underset{\alpha_1+\alpha_2+\alpha_3=2}{=} v_3^+ - (2 - \alpha_1 - \alpha_2)v_0^+ =$$

$$= v_3^+ - 2v_0^+ + \alpha_1 v_0^+ + \alpha_2 v_0^+ \underset{\substack{v_1^- = v_1^+ - \alpha_1 v_0^+ \\ v_2^- = v_2^+ - \alpha_2 v_0^+}}{=} v_3^+ - 2v_0^+ + (v_1^+ - v_1^-) + (v_2^+ - v_2^-) =$$

$$= (v_1^+ + v_2^+ + v_3^+) - 2v_0^+ - v_1^- - v_2^- \underset{v_0^+ = v_1^+ + v_2^+ + v_3^+}{=} v_0^+ - 2v_0^+ - v_1^- - v_2^- = -(v_0^+ + v_1^- + v_2^-)$$

By schematising the equation with a computational scheme, we get



On the left we can see the black box below broken down.

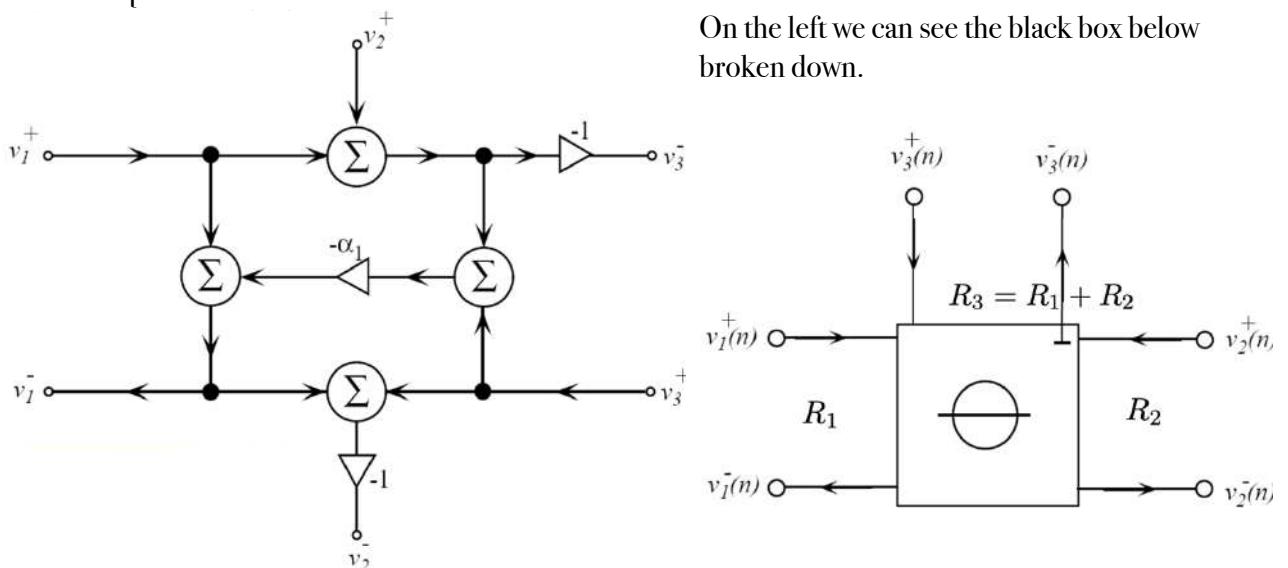
As we can notice, all ports exhibit a local instantaneous reflection! We have to find a way to remove it by turning the series junction into an adaptor. It can be seen as DWG in which  $N = 3$ . We know that it is impossible to have a zero reflection on every port. So we chose one port, and we find the constrain.

$$1 - \alpha_3 = 0 \rightarrow \alpha_1 + \alpha_2 + \alpha_3 = 2 \rightarrow \alpha_2 = 1 - \alpha_1 \rightarrow R_3 = R_1 + R_2$$

In a series junction, choosing one of the port resistances to be equal to the series of the other two port resistances makes that port reflection free (adapted)! We thus obtain

$$\begin{pmatrix} v_1^- \\ v_2^- \\ v_3^- \end{pmatrix} = \begin{pmatrix} 1 - \alpha_1 & -\alpha_1 & -\alpha_1 \\ -\alpha_2 & 1 - \alpha_2 & -\alpha_2 \\ -\alpha_3 & -\alpha_3 & 1 - \alpha_3 \end{pmatrix} \begin{pmatrix} v_1^+ \\ v_2^+ \\ v_3^+ \end{pmatrix} \underset{\substack{\alpha_3=1 \\ \alpha_2=1-\alpha_1}}{=} \begin{pmatrix} 1 - \alpha_1 & -\alpha_1 & -\alpha_1 \\ 1 + \alpha_1 & \alpha_1 & -1 + \alpha_1 \\ -1 & -1 & 0 \end{pmatrix} \begin{pmatrix} v_1^+ \\ v_2^+ \\ v_3^+ \end{pmatrix}$$

Whose computational scheme is



On the left we can see the black box below broken down.

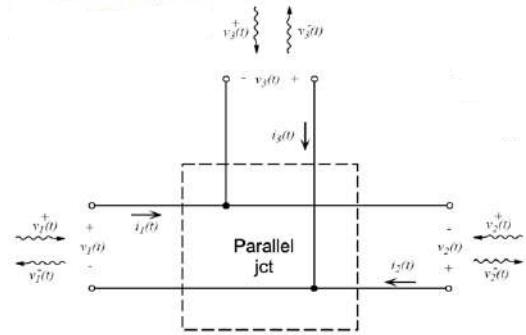
Having applied the constrain, we have removed the instantaneous reflection as  $v_3^-$  does not depend on  $v_3^+$ . The T-edge tells us that the port is adapted.

### Parallel junction

The same but with the admittance.

$$G_k = \frac{1}{R_k}$$

The continuity equations will be



$$v_1 = v_2 = v_3 \rightarrow v_1^+ + v_1^- = v_2^+ + v_2^- = v_3^+ + v_3^-$$

$$i_1 + i_2 + i_3 = 0 \rightarrow G_1(v_1^+ - v_1^-) + G_2(v_2^+ - v_2^-) + G_3(v_3^+ - v_3^-)$$

So let us process as before

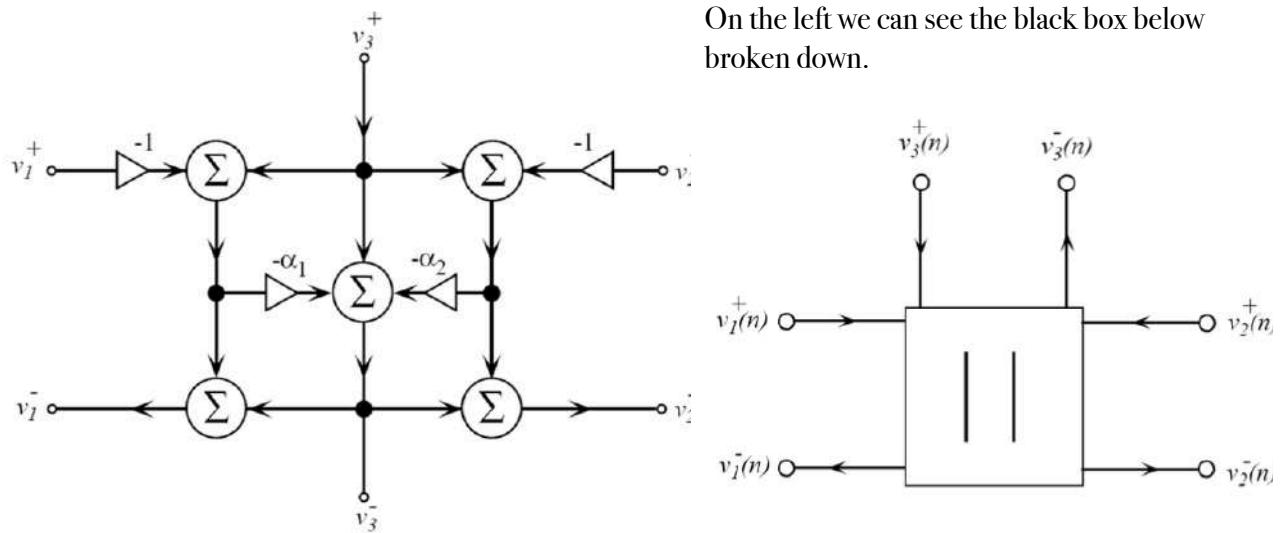
$$\begin{pmatrix} v_1^- \\ v_2^- \\ v_3^- \end{pmatrix} = \underbrace{\left\{ \begin{pmatrix} \alpha_1 & \alpha_1 & \alpha_1 \\ \alpha_2 & \alpha_2 & \alpha_2 \\ \alpha_3 & \alpha_3 & \alpha_3 \end{pmatrix} - \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \right\}}_{\text{Scattering Matrix}} \begin{pmatrix} v_1^+ \\ v_2^+ \\ v_3^+ \end{pmatrix}$$

Inhere,  $\alpha = \frac{2G_i}{G_1+G_2+G_3}$ . Therefore  $1 - \alpha_i$  will be the reflection coefficient. Once again, the three ports are not independent of each other, as  $\alpha_1 + \alpha_2 + \alpha_3 = 2$ . Therefore

$$\begin{aligned} v_1^- &= \sum_{k=1}^3 a_k v_k^+ - v_1^+ & v_1^- &= v_3^+ + (v_3^- - v_1^+) \\ v_2^- &= \sum_{k=1}^3 a_k v_k^+ - v_2^+ & v_2^- &= v_3^+ + (v_3^- - v_2^+) \\ v_3^- &= \sum_{k=1}^3 a_k v_k^+ - v_3^+ & v_3^- &= v_3^+ - \alpha_1(v_3^+ - v_1^-) - \alpha_2(v_3^+ - v_2^-) \end{aligned}$$

By schematising the equation with a computational scheme, we get

On the left we can see the black box below broken down.



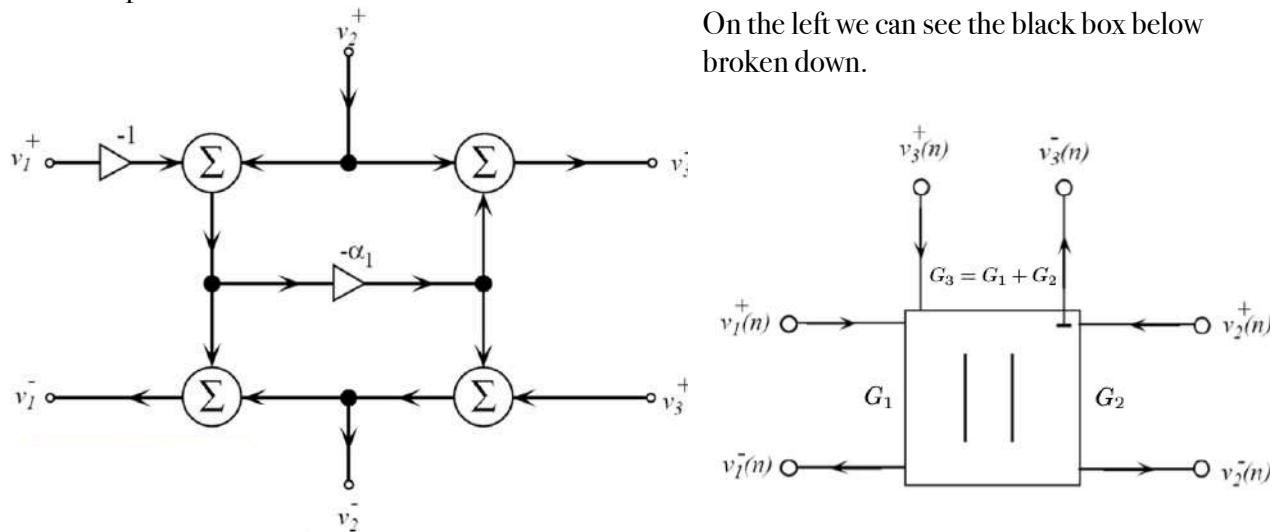
As we can notice, all ports exhibit a local instantaneous reflection! We have to find a way to remove it at least for one of them. Once again, we apply it for port three.

$$1 - \alpha_3 = 0 \rightarrow \alpha_1 + \alpha_2 + \alpha_3 = 2 \rightarrow \alpha_2 = 1 - \alpha_1 \rightarrow G_3 = G_1 + G_2$$

In a parallel junction, choosing one of the port resistances to be equal to the parallel of the other two port resistances makes that port reflection free (adapted)! Thus, what we obtain is

$$\begin{pmatrix} v_1^- \\ v_2^- \\ v_3^- \end{pmatrix} = \begin{pmatrix} \alpha_1 - 1 & \alpha_2 & \alpha_3 \\ \alpha_1 & \alpha_2 - 1 & \alpha_3 \\ \alpha_1 & \alpha_2 & \alpha_3 - 1 \end{pmatrix} \begin{pmatrix} v_1^+ \\ v_2^+ \\ v_3^+ \end{pmatrix} \stackrel{\substack{\alpha_3=1 \\ \alpha_2=1-\alpha_1}}{=} \begin{pmatrix} \alpha_1 - 1 & 1 - \alpha_1 & 1 \\ \alpha_1 & -\alpha_1 & 1 \\ \alpha_1 & 1 - \alpha_1 & 0 \end{pmatrix} \begin{pmatrix} v_1^+ \\ v_2^+ \\ v_3^+ \end{pmatrix}$$

Whose computational scheme is



Having applied the constrain, we have removed the instantaneous reflection as  $v_3^-$  does not depend on  $v_3^+$ . The T-edge tells us that the port is adapted.

What we get is

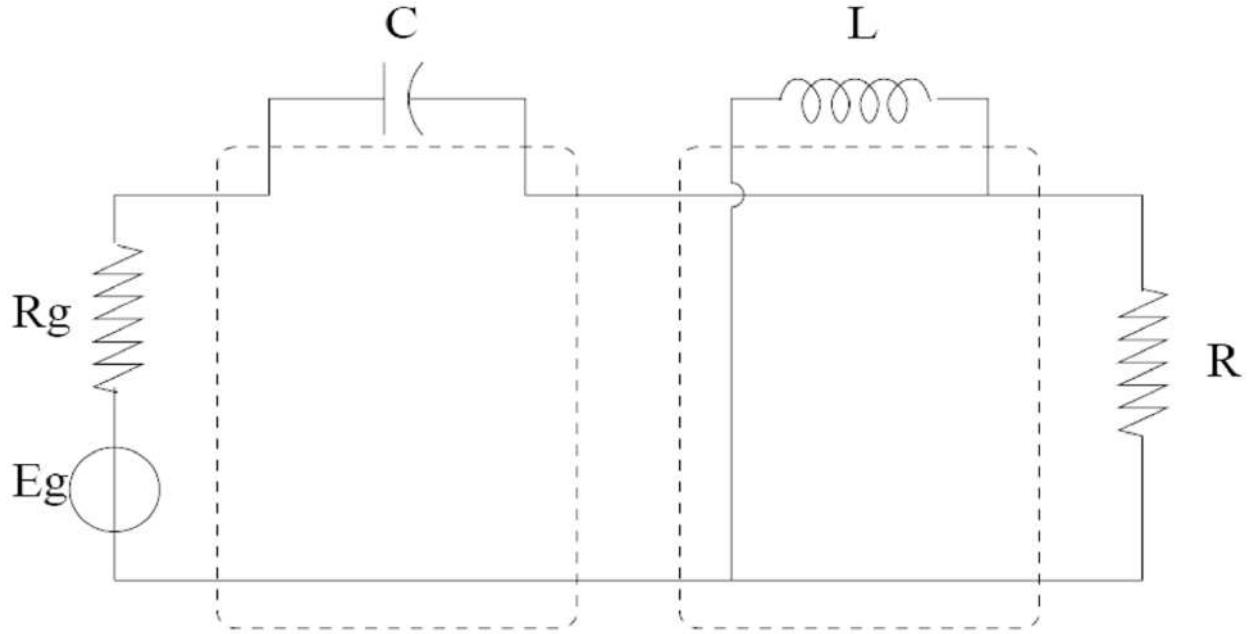
$$G_3 = G_1 + G_2$$

The parallel between the resistances  $R_1$  and  $R_2$  is therefore expressed as

$$\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2}$$

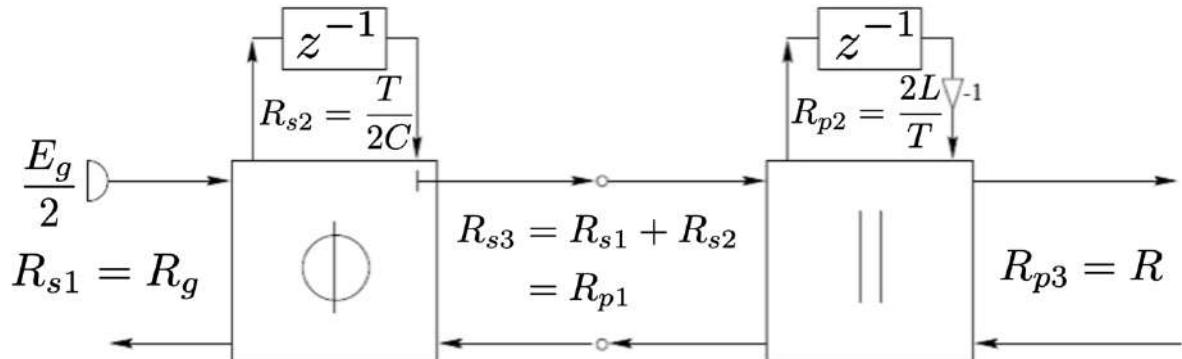
### Modelling an Analogical circuit

We have now all we need to model a linear analogical circuit.

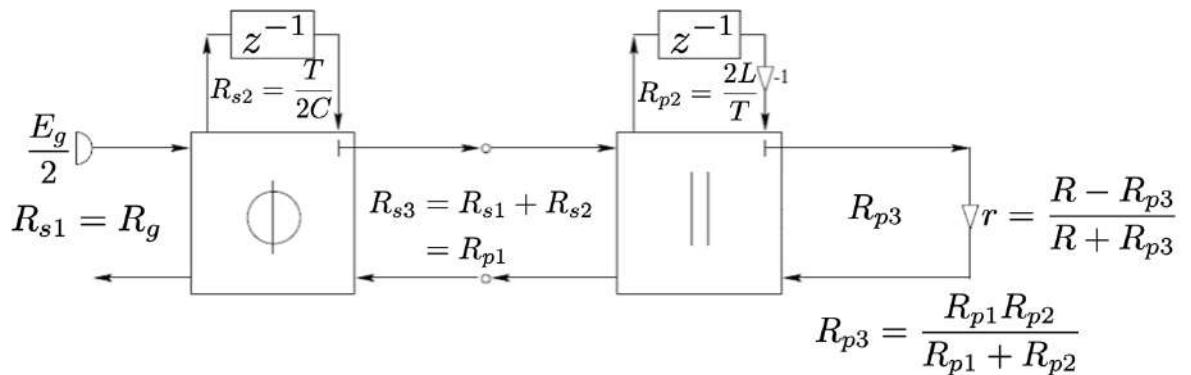


We have two possible choices:

1. Adapt the resistor (the parallel port is reflection free)



2. Adapt the port facing the resistor



We will see how to implement it on Monday with the good Oliver Rocks.

## Modelling NL elements in the WD domain

What happens if we consider non-linear elements into the system? In the most general case, the function is implicit.

$$F_k(v_1, \dots, v_n; i_1, \dots, i_n) = 0 \quad k = 1, \dots, n$$

Let us now recall the definition of waves

$$v = v^+ + v^- \quad i = \frac{v^+ - v^-}{R}$$

and let us plug them into the general equation.

$$F(v, i) = 0 \rightarrow F\left(v^+ - v^-, \frac{v^+ - v^-}{R}\right) = f(v^+, v^-) = 0$$

We need to find the conditions that allow to write the NLE (non-linear expression) in explicit form in the WD (waveDigital) domain.

$$v^- = g(v^+)$$

According to Dini's theorem, we can tell when an implicit function can be explicited.

Reminder: Dini's theorem

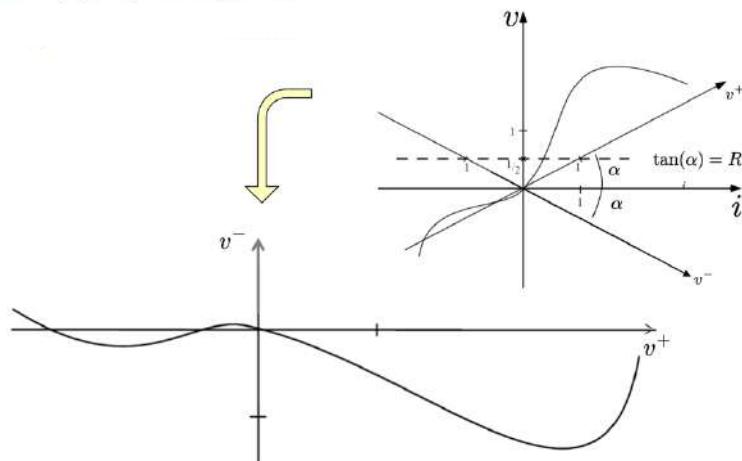
Let  $f(v_0^+, v_0^-)$  be a smooth function. If  $(v_0^+, v_0^-)$  lies on the NL (non linear) characteristics, i.e.  $f(v_0^+, v_0^-) = 0$ , then

$$\frac{\partial f}{\partial v^-} \Big|_{v_0^+, v_0^-} \neq 0 \rightarrow \exists g | f(v_0^+, g(v_0^+)) = 0 \quad \text{around } v_0^+$$

If this condition holds true for all the points of the characteristics, then we have global exploitability.

Anyway, we are interested in the graphical method.

We start from the non-linear characteristics  $F(v, i) = 0$  on the  $K$  plane and we apply the change of variables



Let us see now what happens with a non linear capacitor.

We know the relation between charge and voltage We want to find a way to connect voltage and current. We simply derive the charge.

$$i = \dot{q} = \frac{\partial q}{\partial v} \frac{\partial v}{\partial t} = C(v) \dot{v}$$

Where  $C(v) \triangleq q'(v)$  is the voltage-controlled capacity. It cannot be implemented directly in the WD domain because a non-linear operator and a L-ft cannot be swapped in order. There is a problem: the relation is not linear. We overcome this difficulty by defining two special waves that allow us to treat NLE with memory as if they were memoryless (resistive).

$$\begin{aligned} u^+ &= \frac{1}{2} \left( v + \frac{q}{C} \right) & u^- &= \frac{1}{2} \left( v - \frac{q}{C} \right) \\ v &= u^+ + u^- & q &= C(u^+ - u^-) \end{aligned}$$

Where  $C$  is a “reference” capacity. At this point we have to find the transform that can map voltage and charge into  $v^+$  and  $v^-$ , the variables of the wave digital domain. This is an across integration (as even though we are not working we proper voltage, we are still working with something that is similar to voltage). We simply have to use the current as an intermediate representation between the two domains.

$$\begin{aligned} v &= v^+ + v^- & i &= \frac{v^+ - v^-}{R} & \dot{q} &= i \\ u^+ + u^- &= v^+ + v^- & C(\dot{u}^+ - \dot{u}^-) &= \frac{v^+ - v^-}{R} \end{aligned}$$

We now apply the Laplace transform with  $\tau = RC$  obtaining

$$-V^- + U^+ = V^+ - U^-$$

$$V^- + \tau s U^+ = V^+ + \tau s U^-$$

We now retrieve  $V^-$  and  $U^+$ :

$$\begin{aligned} V^- &= \frac{1 - \tau s}{1 + \tau s} V^+ + \frac{2\tau s}{1 + \tau s} U^- \\ U^+ &= \frac{2}{1 + \tau s} V^+ - \frac{1 - \tau s}{1 + \tau s} U^- \end{aligned}$$

We define then

$$H(s) \triangleq \frac{1 - \tau s}{1 + \tau s} \quad 1 + H(s) = \frac{2}{1 + \tau s} \quad 1 - H(s) = \frac{2\tau s}{1 + \tau s}$$

The variables become

$$V^- = H(s)V^+ + (1 - H(s))U^- \quad U^+ = (1 + H(s))V^+ - H(s)U^-$$

We switch now to the discrete domain

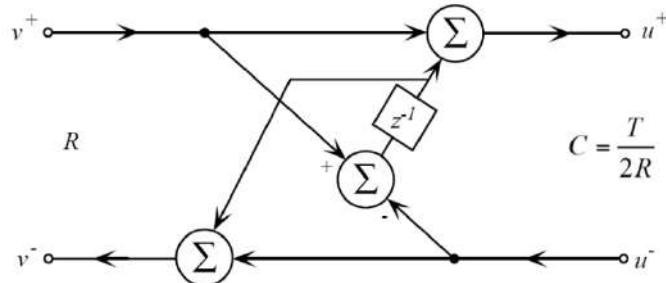
$$s \rightarrow \frac{2}{T} \frac{1 - z^{-1}}{1 + z^{-1}}$$

The transfer function will become

$$\bar{H}(z) \triangleq H(s)|_{s=\frac{21-z^{-1}}{T1+z^{-1}}} = \frac{p + z^{-1}}{1 + pz^{-1}} \quad \text{where } p = \frac{T - 2\tau}{T + 2\tau}$$

We thus obtain something similar to a Kelly-Lochbaum scattering cell whose reflection coefficient is now an all-pass filter. Notice that the output of the filter depends instantaneously on its input unless  $p = 0$ . In order to avoid computability problems, we thus have to choose  $R$  in relation to  $C$  in such a way to obtain  $p = 0$ . So, we impose that  $RC = \frac{T}{2}$  and we get

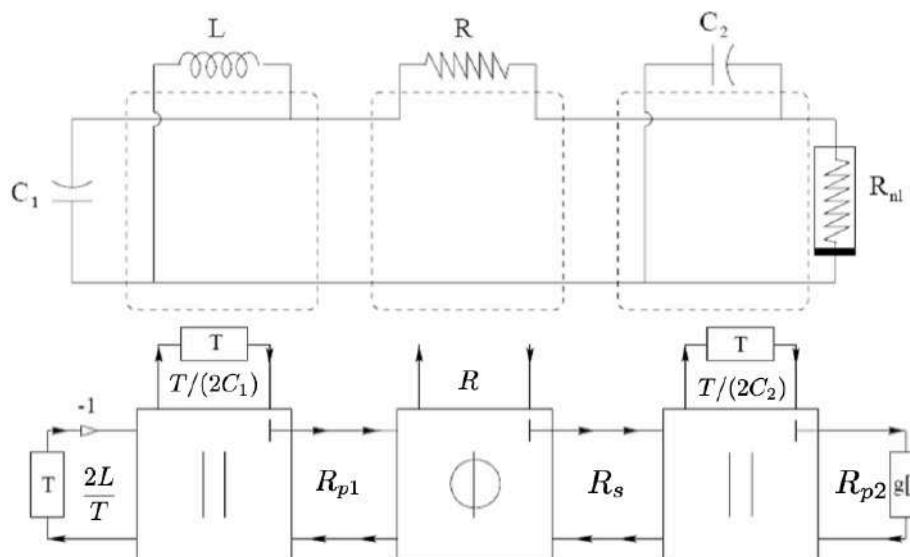
$$\bar{H}(z) = z^{-1}$$



The same is done for the non-linear inductor. The passages will be revised in the next lecture.

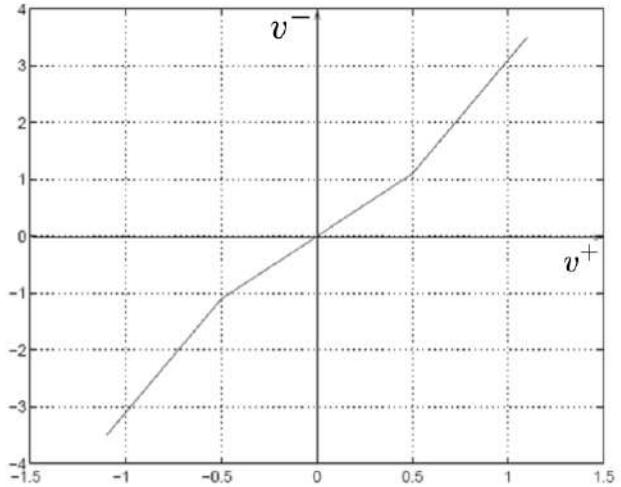
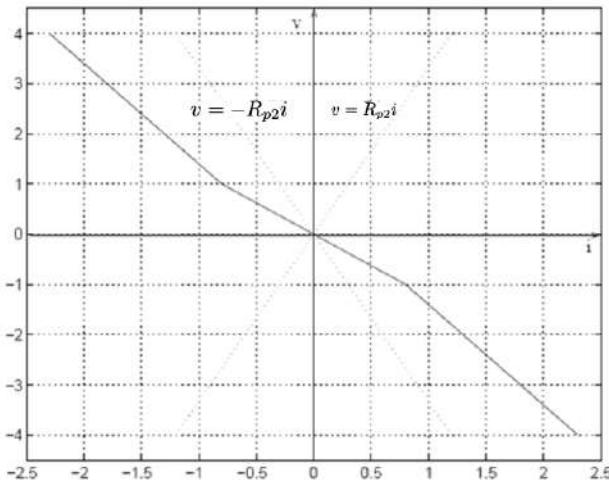
### Example: Chua's circuit

We conclude with an example reporting the Chua's circuit that is a non-linear circuit uses to get a chaotic behaviour.

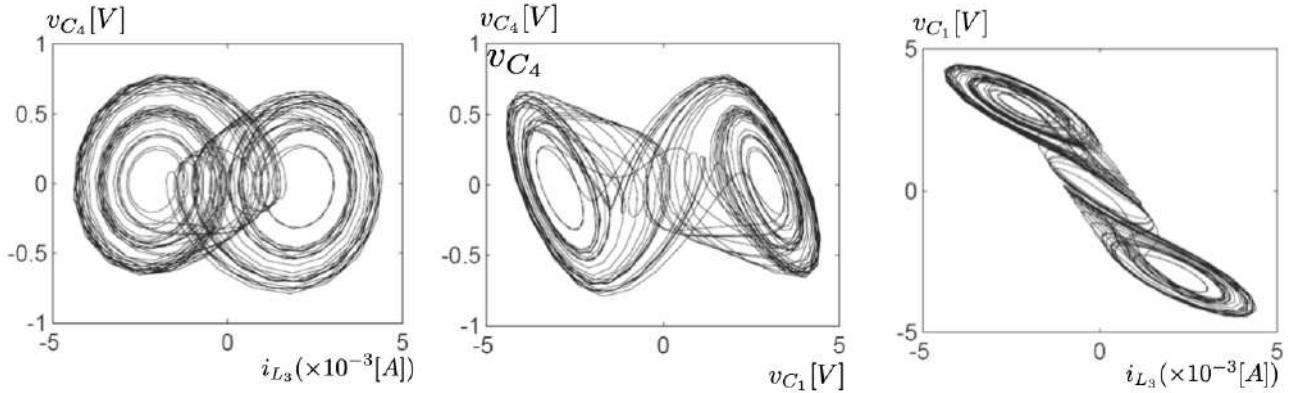


$$R_{p1} = \frac{2L}{T} // \frac{T}{2C_1} = \frac{\frac{L}{C_1}}{\frac{2L}{T} + \frac{T}{2C_1}} \quad R_s = R_{p1} + R \quad R_{p2} = R_s // \frac{T}{2C_2} = \frac{\frac{TR_s}{2C_2}}{R_s + \frac{T}{2C_2}}$$

This is the transformation pf the non-linear resistor

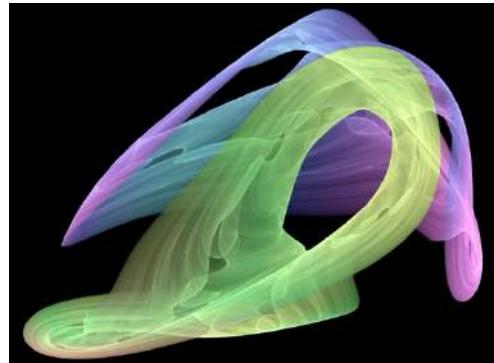


And these orbitations below are the attractors, called double-scroll due to the shape, we get as output



#### Reminder: attractor

In the mathematical field of dynamical systems, an attractor is a set of states toward which a system tends to evolve, for a wide variety of starting conditions of the system. System values that get close enough to the attractor values remain close even if slightly disturbed. In finite-dimensional systems, the evolving variable may be represented algebraically as an n-dimensional vector. The attractor is a region in n-dimensional space. In physical systems, the n dimensions may be, for example, two or three positional coordinates for each of one or more physical entities; in economic systems, they may be separate variables such as the inflation rate and the unemployment rate.



If the evolving variable is two- or three-dimensional, the attractor of the dynamic process can be represented geometrically in two or three dimensions, (as for example in the three-dimensional case depicted to the right). An attractor can be a point, a finite set of points, a curve, a manifold, or even a complicated set with a fractal structure known as a strange attractor (see strange attractor below). If the variable is a scalar, the attractor is a subset of the real number line. Describing the attractors of chaotic dynamical systems has been one of the achievements of chaos theory.

A trajectory of the dynamical system in the attractor does not have to satisfy any special constraints except for remaining on the attractor, forward in time. The trajectory may be periodic or chaotic. If a set of points is periodic or chaotic, but the flow in the neighbourhood is away from the set, the set is not an attractor, but instead is called a Repeller (or Repellor).

## Modelling and implementation of Wave Digital Filters

In today's lecture we will revise and see how to implement what we saw in the previous lectures.

### Circuit Theory

In its long history and evolution, Circuit Theory has had a formidable impact in nearly all fields of engineering. Various lumped linear and nonlinear systems (not only electrical ones) can be represented using equivalent electrical networks. The model of an electrical circuit is made of equation describing the network topology (Kirchhoff voltage laws or loop laws and Kirchhoff current laws or nodes laws) and constitutive equations of circuit elements such as one-port elements (e.g. sources, resistors, capacitors, inductors, diodes) and multi-port elements (e.g. op-amps, transformers, gyrators, transistors, vacuum tubes).

We can characterise systems by means of multivariate systems of ordinary differential equations (ODEs). In order to numerically simulate a circuit, suitable discretisation methods are needed for approximating time derivative in the discrete-time domain. The discretisation makes us bump into a computability problem as when implicit methods are used, the resulting system of discrete-time equations is implicit making the constitutive equations and topological information merge.

Therefore, we need some iterative models able to solve these circuits. The widely adopted solution is given by the Newton-Raphson solvers whose dimensionality roughly equals the number of nodes (or loops) in the circuit. Such a solution is adopted in all the mainstream simulation methods formulated in the Kirchhoff domain, such as:

- Modified Nodal Analysis (MNA) method (SPICE-like software)
- Sparse-Tableau method
- State-Space methods
- Port-Hamiltonian methods



The iteration can be a problem if our application is a real time one. We need to change the kind of paradigm we use. We come up with a theory born during the 70s called Wave Digital Filter (WDF). It was developed by Alfred Fettweis, here on the left, as a methodology for modelling digital filters by discretising reference analogue circuits. It poses the basis for completely new methods for emulating linear and non-linear circuits in the Wave Digital (WD) domain.

Let us see some general considerations on this method:

- A Wave Digital Filter is derived discretising a reference analogue circuit.
- Circuit elements and circuit topology are modelled separately.
- One-port circuit elements are modelled as input-output blocks characterized by scattering relations.
- Topological interconnections of elements are modelled using multi-input-multi-output junctions characterized by scattering matrices.
- Elements and junctions are modelled in a port-wise fashion.
- Each port of an element or junction is characterized by a pair of port variables called wave variables.
- One introduced free parameter per port.
- They can be modelled in a modular fashion. We have a degree of freedom which we can play to avoid implicit equations.

### Definition of voltage waves

We are going now to address all of the wave variables starting from the definition of them. Kirchhoff variables at port  $n$  (of a generic port element of junction) are:

- The port voltage  $v_n$
- The port current  $i_n$

**Wave variables** (voltage waves) are defined as

$$a_n = v_n + Z_n i_n \quad b_n = v_n - Z_n i_n$$

Where:

- $a_n$  is the incident wave
- $b_n$  is the reflected wave
- $Z_n \neq 0$  is a scalar free parameter called *reference port resistance*

The inverse mapping is given by

$$v_n = \frac{a_n + b_n}{2} \quad i_n = \frac{a_n - b_n}{2Z_n}$$

Let us rework the definitions in a vector form

- Kirchhoff-to-wave linear transformation

$$\begin{pmatrix} a_n \\ b_n \end{pmatrix} = \begin{pmatrix} 1 & Z_n \\ 1 & -Z_n \end{pmatrix} \begin{pmatrix} v_n \\ i_n \end{pmatrix}$$

- Wave-to-Kirchhoff linear transformation

$$\begin{pmatrix} v_n \\ i_n \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -\frac{1}{Z_n} \end{pmatrix} \begin{pmatrix} a_n \\ b_n \end{pmatrix}$$

Having this definition of waves, we can shift to one-port elements analysis that in the continuous-time domain ( $t$  is the time variable) are represented by

$$h(v(t), i(t)) = 0$$

where:

- $v(t)$  is the port voltage and  $i(t)$  is the port current
- $h$  is a (linear or nonlinear) dynamic or instantaneous function

In order to pass from continuous time domain, we have nothing to do but sample the continuous signal to obtain its discretised version. By the way, we remind that  $h$  is the constitutive equation (example, for a resistor  $h(v(t), i(t)) = 0 \rightarrow v(t) - Ri(t) = 0$ ). Therefore, in the discrete-time domain

$$\tilde{h}(v[k], i[k]) = 0$$

where:

- $v[k] = v(kT_s)$  and  $i[k] = i(kT_s)$  where  $k$  is the sampling index and  $F_s = \frac{1}{T_s}$  is the sampling frequency.
- If the element is memoryless, we have that  $\tilde{h}(x, y) = h(x, y)$  otherwise  $\tilde{h}(x, y) \neq h(x, y)$ .

The majority of linear one-port elements in the discrete-time domain is characterised by a constitutive equation in the form

$$v[k] = R_e[k]i[k] + V_e[k]$$

Where:

- $R_e[k]$  is a voltage parameter
- $V_e[k]$  is a voltage bias parameter

### Linear resistor

Let us start with the linear resistor whose constitutive equation is given by

$$v(t) = Ri(t)$$

That, in the discrete-time domain becomes

$$v[k] = Ri[k]$$

We notice that is nothing but a special case of the general equation  $v[k] = R_e[k]i[k] + V_e[k]$  where  $R_e = R$  and  $V_e[k] = 0$ .

### Linear resistive voltage generator

In the continuous-time domain the constitutive equation of a linear resistive voltage source with source signal  $V_g(t)$  and internal series resistance  $R_g$  is

$$v(t) = R_g i(t) + V_g(t)$$

That in the discrete-time domain becomes

$$v[k] = R_g i[k] + V_g[k]$$

Once again, we can notice that this last equation is a specific case of the general one in which  $R_e[k] = R_g$  and  $V_e[k] = V_g[k] = V_g(kT_s)$ .

This was everything for static linear memory-less port elements.

## Linear Dynamic Elements

For what it concerns the dynamic elements we have that the constitutive equation of a linear dynamic element (capacitor or inductor) is

$$y(t) = \mu \frac{dx(t)}{dt}$$

Where:

- $x(t)$  is a port voltage or port current
- $y(t)$  is a port current or port voltage
- $\mu$  is a (capacitive or inductive) real coefficient

In the Laplace domain, where  $s$  is the complex frequency variable, the equation becomes

$$Y(s) = s\mu X(s)$$

There are several different methods that allows to approximate a Laplace operation in the discrete domain (where  $z = e^{sT_s}$ ):

- Backward Euler Method

$$s \leftarrow \frac{1 - z^{-1}}{T}$$

- Trapezoidal Rule (a.k.a. bilinear transform or Tustin's method)

$$s \leftarrow \frac{2}{T_s} \frac{1 - z^{-1}}{1 + z^{-1}}$$

- Many other discretisation methods are usable (e.g. finite difference methods, Runge-Kutta methods, etc...)

In order to pass from the Laplace domain to the Fourier's one, we can use the two formal substitution  $s = j\omega$  from which we get

$$\frac{2}{T_s} \frac{1 - z^{-1}}{1 + z^{-1}} \underset{z=e^{sT_s}=e^{j\tilde{\omega}T_s}}{\stackrel{\omega}{=}} \frac{2}{T_s} \frac{1 - e^{-j\tilde{\omega}T_s}}{1 + e^{-j\tilde{\omega}T_s}} = \frac{2}{T_s} \frac{1 - \frac{1}{e^{j\tilde{\omega}T_s}}}{1 + \frac{1}{e^{j\tilde{\omega}T_s}}} = \frac{2}{T_s} \frac{\frac{e^{j\tilde{\omega}T_s} - 1}{e^{j\tilde{\omega}T_s}}}{\frac{e^{j\tilde{\omega}T_s} + 1}{e^{j\tilde{\omega}T_s}}} = \frac{2}{T_s} \frac{e^{j\tilde{\omega}T_s} - 1}{e^{j\tilde{\omega}T_s} + 1}$$

So that

$$j\omega \leftarrow \frac{2}{T_s} \frac{e^{j\tilde{\omega}T_s} - 1}{e^{j\tilde{\omega}T_s} + 1}$$

After making the following simplifications

$$\frac{2}{T_s} \frac{e^{j\tilde{\omega}T_s} - 1}{e^{j\tilde{\omega}T_s} + 1} = \frac{2}{T_s} \frac{e^{j\tilde{\omega}\frac{T_s}{2}} \left( e^{j\tilde{\omega}\frac{T_s}{2}} - e^{-j\tilde{\omega}\frac{T_s}{2}} \right)}{e^{j\tilde{\omega}\frac{T_s}{2}} \left( e^{j\tilde{\omega}\frac{T_s}{2}} + e^{-j\tilde{\omega}\frac{T_s}{2}} \right)} = \frac{2}{T_s} \frac{\frac{\left( e^{j\tilde{\omega}\frac{T_s}{2}} - e^{-j\tilde{\omega}\frac{T_s}{2}} \right)}{2j}}{\frac{\left( e^{j\tilde{\omega}\frac{T_s}{2}} + e^{-j\tilde{\omega}\frac{T_s}{2}} \right)}{2}} 2j \underset{\text{Euler}}{\stackrel{\omega}{=}} \frac{2}{T_s} \frac{\sin\left(\tilde{\omega}\frac{T_s}{2}\right)j}{\cos\left(\tilde{\omega}\frac{T_s}{2}\right)}$$

We get

$$j\omega \leftarrow j \frac{2}{T_s} \tan\left(\tilde{\omega}\frac{T_s}{2}\right)$$

According to the last equation, it is possible to express in closed-form the reference “continuous-time frequency”  $\omega$  as a function of the “discrete-time frequency”  $\tilde{\omega}$  using the warping mapping

$$\omega = \frac{2}{T_s} \tan\left(\tilde{\omega} \frac{T_s}{2}\right)$$

If  $\tilde{\omega}$  is small, we get that  $\omega = \frac{2}{T_s} \tan\left(\tilde{\omega} \frac{T_s}{2}\right) \cong \frac{2}{T_s} \tilde{\omega} \frac{T_s}{2} = \tilde{\omega}$ . So, they differ more and more at high frequencies. The higher the sampling frequency  $F_s = \frac{1}{T_s}$ , the more the difference between  $\omega$  and  $\tilde{\omega}$  becomes negligible in the whole frequency range of interest.

Let us see now some important constitutive equations for some components.

### Linear Capacitor

The constitutive equation of the linear capacitor can be written as  $i(t) = C \frac{dv(t)}{dt}$ , that in the Laplace domain becomes

$$I(s) = sCV(s)$$

After applying the bilinear transform  $s \leftarrow \frac{2}{T_s} \frac{1-z^{-1}}{1+z^{-1}}$  to the equation we get

$$v[k] = \frac{T_s}{2C} i[k] + \frac{T_s}{2C} i[k-1] + v[k-1]$$

Once again, can be seen as a special case of the general equation in which

- $R_e[k] = \frac{T_s}{2C}$
- $V_e[k] = \frac{T_s i[k-1]}{2C} + v[k-1]$

### Linear Inductor

The same considerations can be done for the linear inductor, whose equation is

$$v(t) = L \frac{di(t)}{dt}$$

In the Laplace domain

$$V(s) = sLI(s)$$

After applying the bilinear transform  $s \leftarrow \frac{2}{T_s} \frac{1-z^{-1}}{1+z^{-1}}$  to the equation we get

$$v[k] = \frac{2L}{T_s} i[k] + \frac{2L}{T_s} i[k-1] - v[k-1]$$

Once again, can be seen as a special case of the general equation in which

- $R_e[k] = \frac{2L}{T_s}$
- $V_e[k] = \frac{2Li[k-1]}{T_s} - v[k-1]$

### Linear Wave Digital One-Port Element

We are still in the Kirchhoff domain! To pass into the wave domain we have to apply the scattering matrix. We know that

$$v[k] = \frac{a[k] + b[k]}{2} \quad i[k] = \frac{a[k] - b[k]}{2Z[k]}$$

That we plug into the general equation

$$v[k] = R_e[k]i[k] + V_e[k]$$

Obtaining

$$\frac{a[k] + b[k]}{2} = R_e[k] \frac{a[k] - b[k]}{2Z[k]} + V_e[k]$$

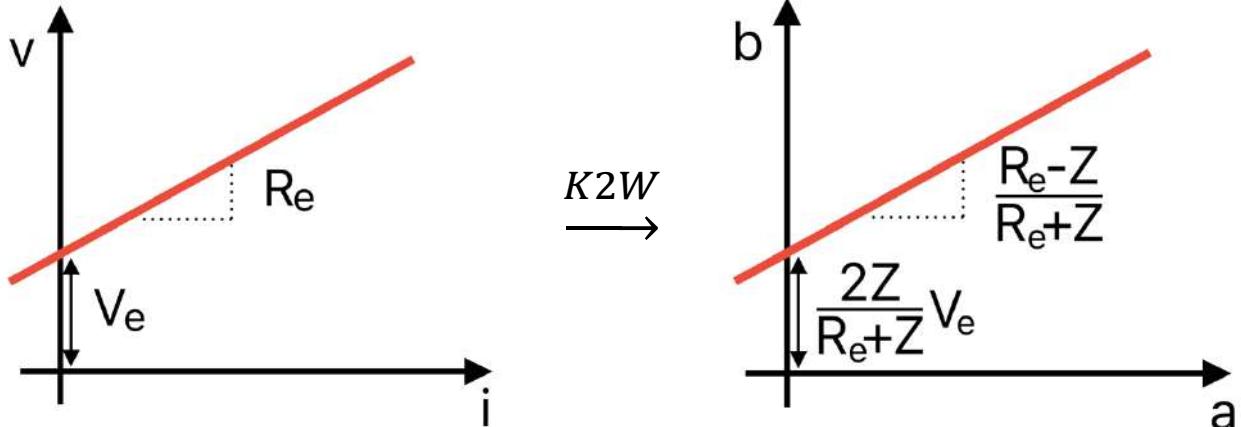
That solved for  $b[k]$  returns the scattering relation of a generic linear one-port element

$$b[k] = \frac{R_e[k] - Z[k]}{R_e[k] + Z[k]} a[k] + \frac{2Z[k]}{R_e[k] + Z[k]} V_e[k]$$

The adaptation case (in which the instantaneous dependency of  $b[k]$  from  $a[k]$  is eliminated) is

$$b[k] = V_e[k] \quad \text{with } Z[k] = R_e[k]$$

Graphically, when we pass from the Kirchhoff domain to the wave one, we get



We remark that  $Z[k]$  is artificially set to obtain numerical advantages: it does not come from the equations!

By summarising the equations, we have

Constitutive Equation	Wave Mapping	Adaptation Condition
$v(t) = V_g(t) + R_g i(t)$	$b[k] = V_g[k]$	$Z[k] = R_g$
$v(t) = R i(t)$	$b[k] = 0$	$Z[k] = R$
$i(t) = C \frac{dv(t)}{dt}$	$b[k] = a[k - 1]$	$Z[k] = \frac{T_s}{2C}$
$v(t) = L \frac{di(t)}{dt}$	$b[k] = -a[k - 1]$	$Z[k] = \frac{2L}{T_s}$

We notice that none of these equations depends on the previous values of the incident wave.

## Nonlinear elements

Times to deal with non-linear elements. We start from the diode whose equation is given by the Shockley diode model

$$i(t) = I_s \left( e^{\frac{v(t)}{\eta V_{Th}}} - 1 \right)$$

Where:

- $I_s$  is the saturation current
- $V_{Th}$  the thermal voltage
- $\eta$  the ideality factor

Since the equation is not linear, it cannot be represented with the general equation we saw before.

What we can do, instead, is to obtain its discrete-time version by applying the definition of wave variables obtaining a transcendental equation in the WD domain.

$$\frac{a[k] - b[k]}{2Z[k]} = I_s \left( e^{\frac{a[k]+b[k]}{\eta V_{Th}}} - 1 \right)$$

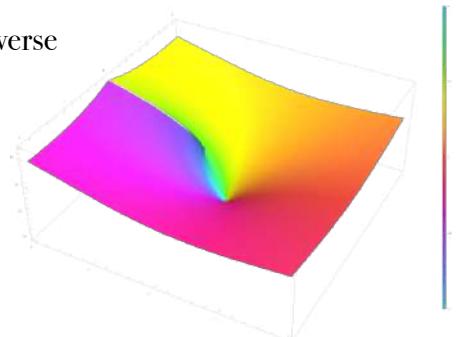
This is what we get after some passages

$$b[k] = a[k] + 2Z[k]I_s - 2\eta V_{th}W\left(\frac{Z[k]I_s}{\eta V_{th}}e^{\frac{Z[k]I_s+a[k]}{\eta V_{th}}}\right)$$

### Reminder: Lambert Function

The Lambert  $W$  function, also called the omega function or product logarithm, is a multivalued function, namely the branches of the converse relation of the function  $f(w) = we^w$ , where  $w$  is any complex number and  $e^w$  is the exponential function. The function is named after Johann Lambert, who considered a related problem in 1758. Building on Lambert's work, Leonhard Euler described the  $W$  function per se in 1783. As we can see, it is implicitly defined as

$$x = W(x)e^{W(x)}$$



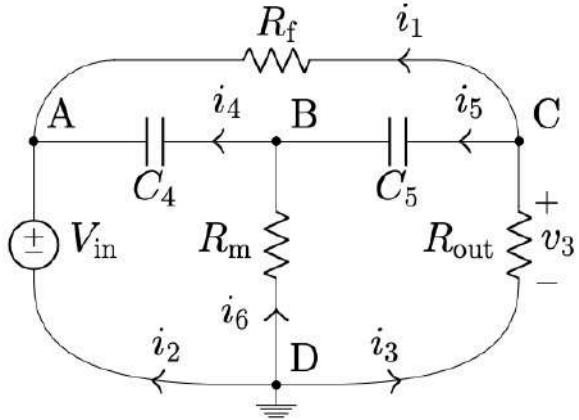
Since the Lambert function that appears in the diode equation is obviously non-linear, it cannot be adapted!

### Modelling the topology

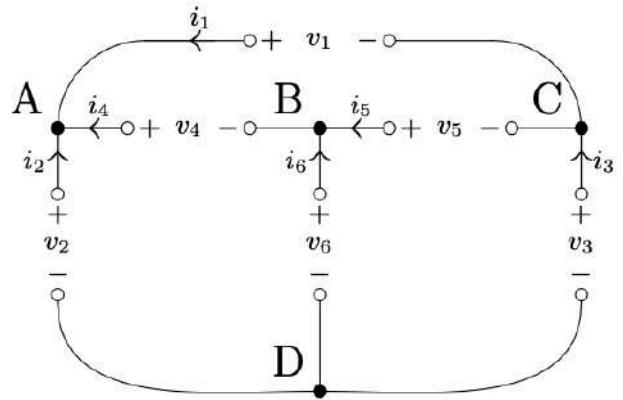
We have now to model the topology that embeds all the consideration that we have done on the Kirchhoff laws. A  $N$ -port topological junction is an open interconnection network (i.e., without electrical loads) characterised by:

- A vector of port voltages  $\underline{v} = [v_1, \dots, v_N]^T$
- A vector of port currents  $\underline{i} = [i_1, \dots, i_N]^T$

For example:



(a) Reference circuit.



(b) Topological connection network.

If we look at a network, we can find a sub-set of independent port voltages so that

$$\underline{v} = \underline{\underline{Q}}^T \underline{v}_t$$

Where:

- $\underline{v}_t$  is the vector of size  $q \times 1$  collecting independent port voltages
- $\underline{\underline{Q}}$  is the **fundamental cut-set matrix** of size  $q \times N$

We can also find a subset of independent port currents such that

$$\underline{i} = \underline{\underline{B}}^T \underline{i}_l$$

Where:

- $\underline{i}_l$  is the vector of size  $p \times 1$  collecting independent port currents
- $\underline{\underline{B}}$  is the **fundamental loop matrix** of size  $p \times N$

The two following properties must holds:

- $p + q = N$
- The orthogonality property

$$\underline{\underline{B}} \underline{\underline{Q}}^T = \underline{0}_{p \times q} \quad \underline{\underline{Q}}^T \underline{\underline{B}} = \underline{0}_{q \times p}$$

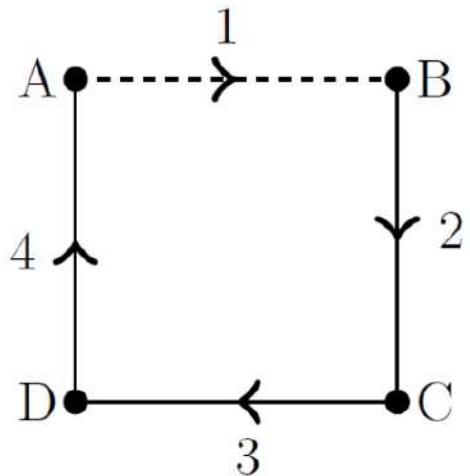
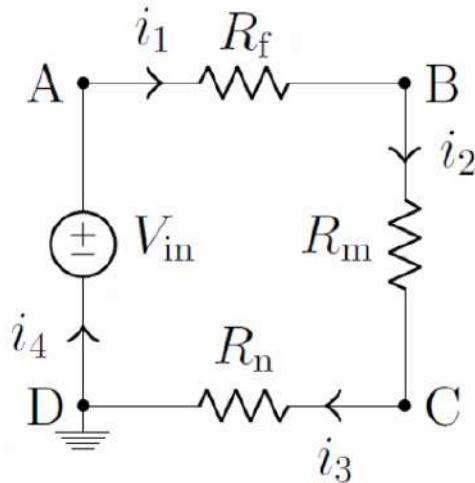
But, how do we generate independent port variables?

In order to answer to this question, we have to abstract farther by introducing the digraph, a directing graph. We consider digraph  $\mathcal{D}$  of the reference circuit where the edges represent the loads of the connection network (one per port), while the vertices represent the nodes of the circuit. By applying a *tree-cotree decomposition* to  $\mathcal{D}$  we get

- A *tree*  $\mathcal{T}$  of  $\mathcal{D}$  defined as a connected acyclic subgraph of  $\mathcal{D}$  containing all vertices
- A *cotree*  $\mathcal{C}$  of  $\mathcal{D}$  defined as a subgraph of  $\mathcal{D}$  containing all the edges of  $\mathcal{D}$  that are not in a reference tree  $\mathcal{T}$ .

Hence, independent port voltages in  $\underline{v}_t$  are those related to the edges of the tree while independent port currents in  $\underline{i}_l$  are those related to the edges of the cotree.

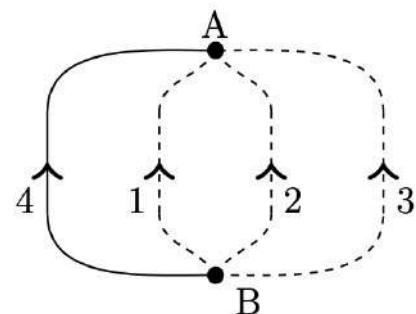
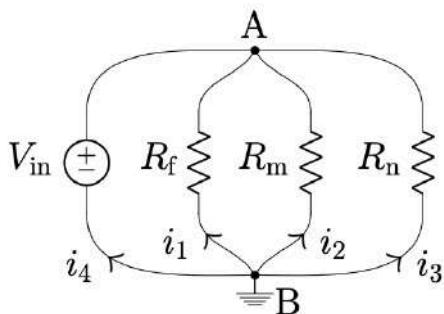
### *Example 1: series connection network*



The dashed line is indicating the cotree. The equations will be

$$\underline{i} = \underline{\underline{B}}^T \underline{i}_l \quad \rightarrow \quad \begin{pmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} i_1$$

### *Example 2: Parallel*

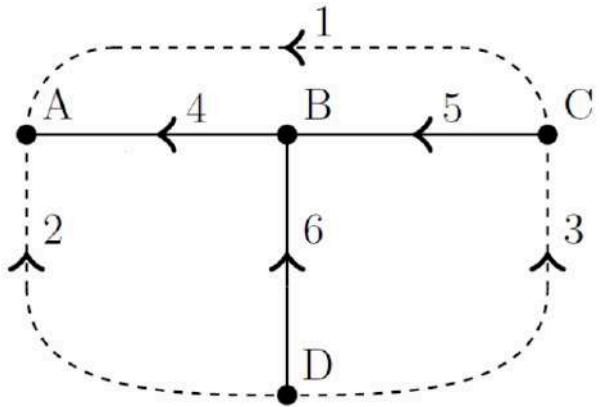
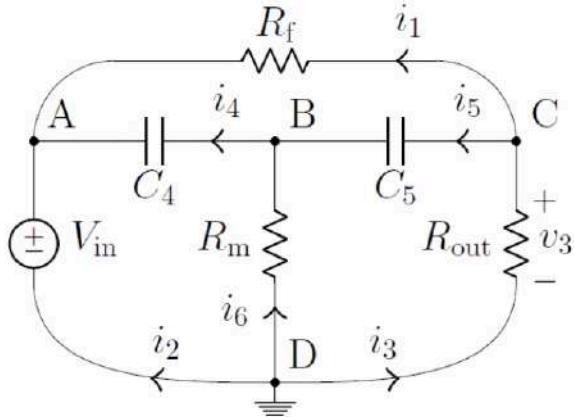


For which we have

$$\underline{v} = \underline{\underline{Q}}^T \underline{v}_l \quad \rightarrow \quad \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} v_4$$

### Example 3: Bridged-Tee Connection Network

This can be applied to any kind of connection network.



$$\underline{i} = \underline{\underline{B}}^T \underline{i}_I \quad \rightarrow \quad \begin{pmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \\ i_5 \\ i_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & -1 \end{pmatrix} \begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix}$$

### WD Junctions (Adaptors)

In the WD domain a topological connection network is modelled as a WD scattering junction (also called adaptor). Kirchhoff-to-Wave mapping of port variables

$$\underline{v} = \frac{1}{2}(\underline{a} + \underline{b}) \quad \underline{i} = \frac{1}{2}\underline{\underline{Z}}^{-1}(\underline{a} - \underline{b})$$

- $\underline{a} = (a_1, \dots, a_N)^T$  vector of waves incident to the junction
- $\underline{b} = (b_1, \dots, b_N)^T$  vector of waves reflected by the junction
- $\underline{\underline{Z}} = \text{diag}(Z_1, \dots, Z_N)$  is the diagonal matrix of free parameters

The scattering equation allows us to express the reflected waves vector in function of the incident wave vector.

$$\underline{b} = \underline{\underline{S}} \underline{a}$$

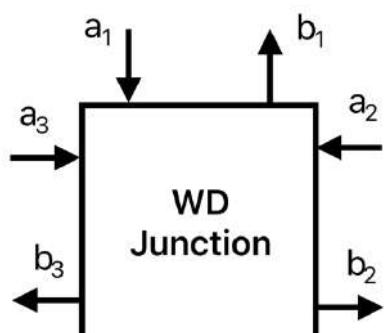
Where  $\underline{\underline{S}}$  is a  $N \times N$  scattering matrix

Each port will have its incident and reflected wave as we can see in the scheme on the right.

But how can we form this scattering matrix?

If  $q \leq p$  we use

$$\underline{\underline{S}} = 2\underline{\underline{Q}}^T \left( \underline{\underline{Q}} \underline{\underline{Z}}^{-1} \underline{\underline{Q}}^T \right)^{-1} \underline{\underline{Q}} \underline{\underline{Z}}^{-1} - \underline{\underline{I}}$$



Where:

- $\underline{\underline{I}}$  is the  $N \times N$  identity matrix
- The inversion of the  $q \times q$  matrix  $\underline{\underline{Q}} \underline{\underline{Z}}^{-1} \underline{\underline{Q}}^T$  is required

If  $q \geq p$  we use

$$\underline{\underline{S}} = \underline{\underline{I}} - 2\underline{\underline{Z}} \underline{\underline{B}}^T \left( \underline{\underline{B}} \underline{\underline{Z}} \underline{\underline{B}}^T \right)^{-1} \underline{\underline{B}}$$

Where:

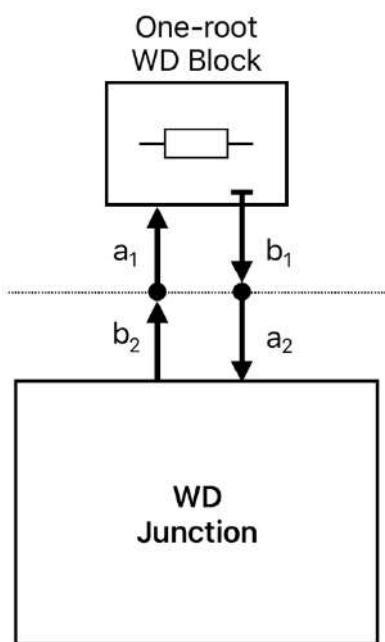
- $\underline{\underline{I}}$  is the  $N \times N$  identity matrix
- The inversion of the  $p \times p$  matrix  $\underline{\underline{B}} \underline{\underline{Z}} \underline{\underline{B}}^T$  is required

As we did for the elements, also for wave digital junctions we can perform adaptations. This means that can be made reflection-free. The  $n$ th port of a WD junction is made reflection-free if the  $n$ th diagonal entry  $s_{nn}$  of  $\underline{\underline{S}}$  is imposed to be zero ( $s_{nn} = 0$ ). This condition can be satisfied by properly setting the free parameter  $Z_n$ . Examples:

- The  $n$ th port of a  $N$ -port series WD junction is made reflection-free by setting  $Z_n = \sum_{k \neq n} Z_k$
- The  $n$ th port of a  $N$ -port series WD junction is made reflection-free by setting  $Z_n^{-1} = \sum_{k \neq n} Z_k^{-1}$

### Connection tree Structures

But, why adaptation? Let us draw a simple scheme to understand it.



We can perform a connection between the two elements just by putting  $a_1 = b_2$  and  $a_2 = b_1$ . According to the scattering relation we can write that

$$b_1[k] = f(a_1[k]) = f(b_2[k]) = f(g[a_2[k]]) \underset{a_2=b_1}{=} f(g[b_1[k]]).$$

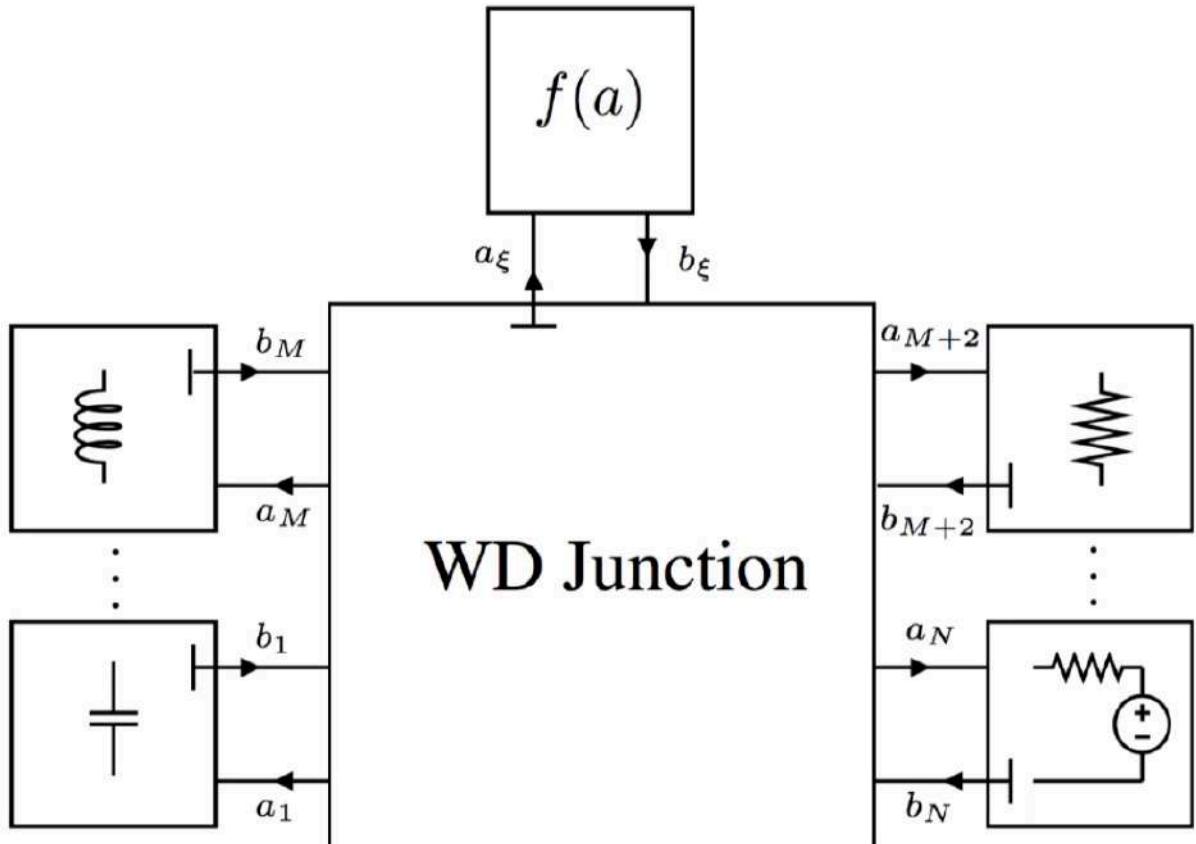
Oh no!  $b_1$  depends on  $b_1$ . This means that we cannot compute  $b_1$  by means of a close formula.

To remove this awful dependence, we introduce adaptation imposing  $b_1 = f(a_1)$ . In this case it will be possible to perform the connection and write a close expression to find  $b_1$ .

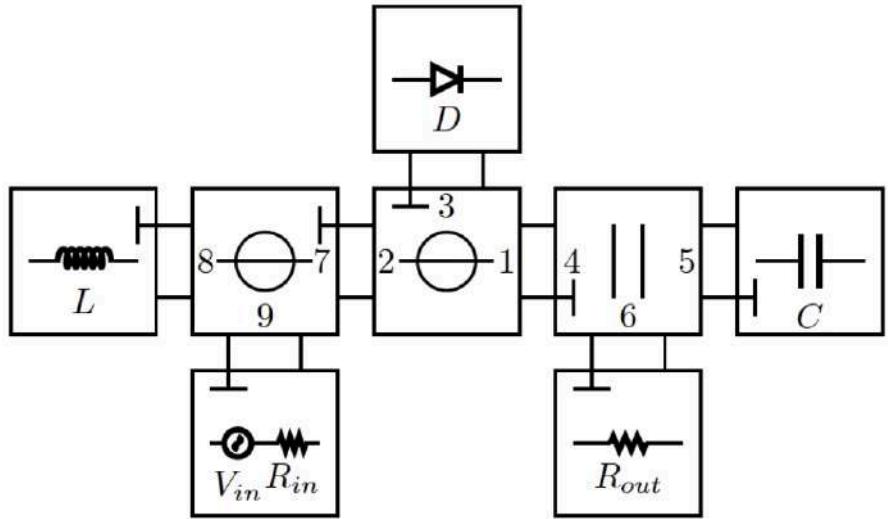
Let us see now how to arrange what we have seen so far into structures. We look now at the class of circuits that contains at most one non-linear element. These circuits can be organised into the wave digital domain by means of **connection tree** whose **root** is the nonlinear one-port element (the element that cannot be adapted). The WD topological junctions are

the **nodes** of the circuit: ports of WD junctions either connected to other WD junctions or to the nonlinear element are made reflection-free. Pay attention to the fact that the last sentence embeds a constrain that we have to satisfy! The port facing upwards needs to be adapted. Then, all the linear WD one-port elements are the **leaves** that are all adapted. In case the topology is solely made of series-parallel connections, the WDF can be modelled as a **Binary Connection Tree (BCT)**.

Let us see an example: a generic connection tree with one node.



The adaptation is described by a line inside the junction. As we can notice, all the leaves are adapted. The upward facing port is adapted from the side of the junction. Let us see now how to implement a binary connection tree where, as we can see on the right, all the nodes are 3-port series/parallel adaptors. Once again, we notice that the upward facing port needs to be adapted.



## Computational flow in Connection Trees

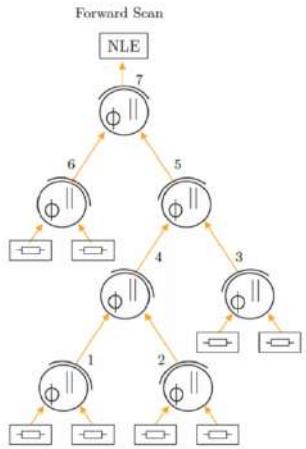
At this point we need a computational flow that allows us to simulate in discrete time the wave digital filter itself. It can be performed iterating the following steps:

### 1. Forward scan

We follow the inverse path from the leaves to the root “connecting the ports”. It will be constituted by the computation of the reflected waves from the linear elements which, since are adapted, do not depend on the incident wave. Basically, we have to compute the reflected wave of each junction at the port corresponding to the upward facing port.

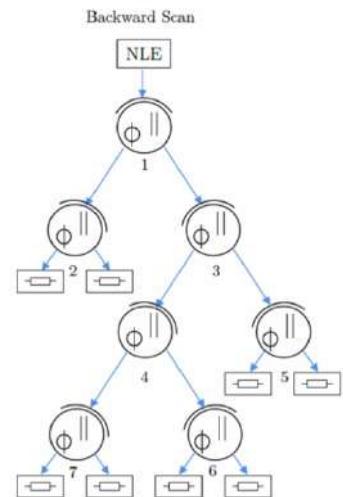
### 2. Local non-linear scattering stage at the root

Once we reached the root, we have to compute the reflected wave of the non-linear element based on the incident wave we have computed in the previous point. Pay attention to the dichotomy in notation that incident and reflected waves with respect to the elements are going to be reflected and incident wave with respect to the junction.



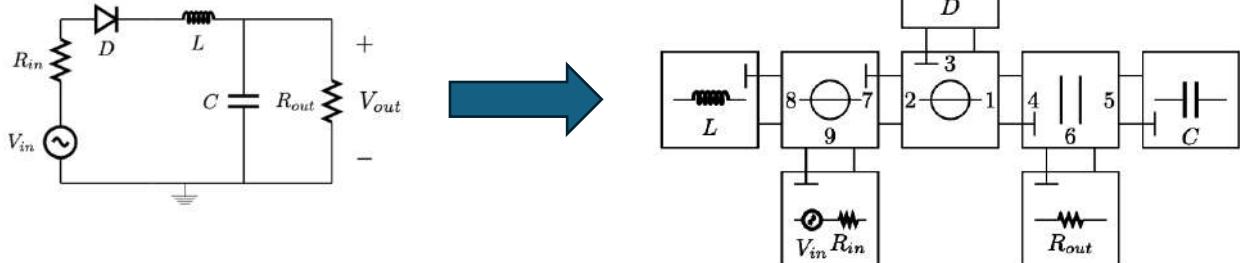
### 3. Backward scan

We connect the ports following the pathway that goes from the root to the leaves. On the right, there are some pictures that can explain us more clearly what it happens. Level by level we reach the non-linear element. Once we reached it, we propagate the reflected waves towards the tree.



## Example

Let us see a practical application, for example, let us try to analyse the envelope follower circuit. The envelop follower circuit returns the value of the amplitude of a signal.



The Wave Digital filters is composed by four leaves (the inductor, the capacitor, the  $R_{out}$  and the series between the voltage generator  $V_{in}$  and  $R_{in}$ ), three junctions (two series adaptors and one parallel adaptor) and a root (the diode). In order not to have too many indexes, in general only the ports are numerated and not all the connections.

We are ready to perform the port connections.

The port connection between port number 1 of the series adaptor and port number 4 of the parallel adaptor on the right is performed imposing the following constraints.

$$a_1[k] = b_4[k] \quad a_4[k] = b_1[k] \quad Z_1 = Z_4$$

Similarly, connection between port number 2 of the series adaptor and port number 7 of the series adaptor on the left is performed imposing the following constraints.

$$a_2[k] = b_7[k] \quad a_7[k] = b_2[k] \quad Z_2 = Z_7$$

And so on and so forth.

Then we have to impose the adaptation conditions. A WD block with a T-shaped symbol at a port is adapted at that port (that port is reflection-free). For instance, the parallel adaptor is adapted at port number 4 and all one-port WD elements are adapted, except for the diode which cannot be adapted. In this specific case, the adaptation conditions set at ports facing linear elements are

$$Z_9 = R_{in} \quad Z_6 = R_{out} \quad Z_5 = \frac{T_s}{2C} \quad Z_8 = \frac{2L}{T_s}$$

And adaptation conditions set at port facing other adaptors are

$$Z_1 = Z_4 = \frac{Z_5 Z_6}{Z_5 + Z_6} \quad Z_2 = Z_7 = Z_8 + Z_9 \quad Z_3 = Z_1 + Z_2$$

Then we have to write the scattering relations of the elements

Inductor with inductance  $L$       Resistor with resistance  $R_{out}$       Capacitor with capacitance  $C$

$$a_8[k] = -b_8[k-1] \quad a_6[k] = 0 \quad a_5[k] = b_5[k-1]$$

Voltage source  $V_{in}$  with series resistance  $R_{in}$       Diode  $D$

$$a_9[k] = V_{in}[k] \quad a_3[k] = b_3[k] + 2Z_3I_s - 2\eta V_{th}W \left( \frac{Z_3I_s}{\eta V_{th}} e^{\frac{Z_3I_s+b_3[k]}{\eta V_{th}}} \right)$$

And the scattering relation of the WD junctions

Series adaptor with ports 1, 2, 3  
and scattering matrix  $\underline{\underline{S}}_{S_1}$

$$\begin{pmatrix} b_1[k] \\ b_2[k] \\ b_3[k] \end{pmatrix} = \underline{\underline{S}}_{S_1} \begin{pmatrix} a_1[k] \\ a_2[k] \\ a_3[k] \end{pmatrix}$$

Where

$$\begin{aligned} \underline{\underline{S}}_{S_1} &= \begin{pmatrix} s_{11} & s_{12} & s_{13} \\ s_{21} & s_{22} & s_{23} \\ s_{31} & s_{32} & s_{33} \end{pmatrix} = \\ &= \begin{pmatrix} \gamma_{S_1} & (\gamma_{S_1} - 1) & (\gamma_{S_1} - 1) \\ -\gamma_{S_1} & (1 - \gamma_{S_1}) & -\gamma_{S_1} \\ -1 & -1 & 0 \end{pmatrix} \end{aligned}$$

where, in turns,

$$\gamma_{S_1} = \frac{Z_2}{Z_1 + Z_2}$$

Series adaptor with ports 7, 8, 9  
and scattering matrix  $\underline{\underline{S}}_{S_2}$

$$\begin{pmatrix} b_7[k] \\ b_8[k] \\ b_9[k] \end{pmatrix} = \underline{\underline{S}}_{S_2} \begin{pmatrix} a_7[k] \\ a_8[k] \\ a_9[k] \end{pmatrix}$$

Where

$$\begin{aligned} \underline{\underline{S}}_{S_2} &= \begin{pmatrix} s_{77} & s_{78} & s_{79} \\ s_{87} & s_{88} & s_{89} \\ s_{97} & s_{98} & s_{99} \end{pmatrix} = \\ &= \begin{pmatrix} 0 & -1 & -1 \\ -\gamma_{S_2} & (1 - \gamma_{S_2}) & -\gamma_{S_2} \\ (\gamma_{S_2} - 1) & (\gamma_{S_2} - 1) & \gamma_{S_2} \end{pmatrix} \end{aligned}$$

where, in turns,

$$\gamma_{S_2} = \frac{Z_8}{Z_7 + Z_9}$$

Parallel adaptor with ports 4, 5, 6  
and scattering matrix  $\underline{\underline{S}}_{P_1}$

$$\begin{pmatrix} b_4[k] \\ b_5[k] \\ b_6[k] \end{pmatrix} = \underline{\underline{S}}_{P_1} \begin{pmatrix} a_4[k] \\ a_5[k] \\ a_6[k] \end{pmatrix}$$

Where

$$\begin{aligned} \underline{\underline{S}}_{P_1} &= \begin{pmatrix} s_{44} & s_{45} & s_{46} \\ s_{54} & s_{55} & s_{56} \\ s_{64} & s_{65} & s_{66} \end{pmatrix} = \\ &= \begin{pmatrix} 0 & (1 - \gamma_{P_1}) & \gamma_{P_1} \\ 1 & \gamma_{P_1} & \gamma_{P_1} \\ 1 & (1 - \gamma_{P_1}) & (\gamma_{P_1} - 1) \end{pmatrix} \end{aligned}$$

where, in turns,

$$\gamma_{P_1} = \frac{Z_5}{Z_4 + Z_6}$$

Maybe it is easier to see these relations directly into the algorithm.

### Forward scan

Firstly, we write the wave reflected from the linear elements. Pay attention to write them with respect to the junction and not with respect to the element! As we notice, in fact, we have  $a_x = \dots$  and not  $b_x = \dots$ . And we will have

$$a_9[k] = V_{in}[k]$$

$$a_6[k] = 0$$

$$a_5[k] = b_5[k - 1]$$

$$a_8[k] = -b_8[k - 1]$$

Voltage source

We remind that the adaptive scattering relation of a resistor is simply zero

Capacitor. We notice that depends on the previous sample.

Inductor. This time again, it depends on the previous sample.

Once we compute these waves, we have to propagate them upward into the connection tree. This is done simply computing the reflective wave at one specific point.

We start from the first layer of adaptors

At port 4, we multiply the line of the scattering matrix corresponding to that port to the line wave impinging to it. Therefore, we will get

$$b_4[k] = (1 - \gamma_{P_1})a_5[k] + \gamma_{P_1}a_6[k]$$

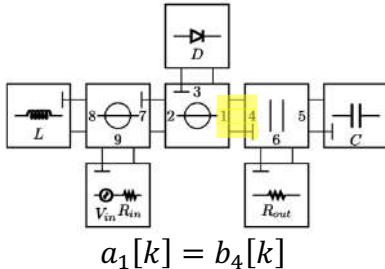
$$\begin{pmatrix} b_4[k] \\ b_5[k] \\ b_6[k] \end{pmatrix} = \begin{pmatrix} 0 & (1 - \gamma_{P_1}) & \gamma_{P_1} \\ 1 & \gamma_{P_1} & \gamma_{P_1} \\ 1 & (1 - \gamma_{P_1}) & (\gamma_{P_1} - 1) \end{pmatrix} \begin{pmatrix} a_4[k] \\ a_5[k] \\ a_6[k] \end{pmatrix}$$

The same is done for port 7

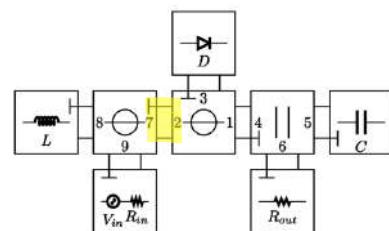
$$b_7[k] = -a_8[k] - a_9[k]$$

$$\begin{pmatrix} b_7[k] \\ b_8[k] \\ b_9[k] \end{pmatrix} = \begin{pmatrix} 0 & -1 & -1 \\ -\gamma_{S_2} & (1 - \gamma_{S_2}) & -\gamma_{S_2} \\ (\gamma_{S_2} - 1) & (\gamma_{S_1} - 1) & \gamma_{S_2} \end{pmatrix} \begin{pmatrix} a_7[k] \\ a_8[k] \\ a_9[k] \end{pmatrix}$$

At this point, we compute the waves reflected from the second layer of adaptors always looking at the scattering matrixes and at the connections.



$$a_1[k] = b_4[k]$$



$$a_2[k] = b_7[k]$$

$$b_3[k] = -a_1[k] - a_2[k]$$

$$\begin{pmatrix} b_1[k] \\ b_2[k] \\ b_3[k] \end{pmatrix} = \begin{pmatrix} \gamma_{S_1} & (\gamma_{S_1} - 1) & (\gamma_{S_1} - 1) \\ -\gamma_{S_1} & (1 - \gamma_{S_1}) & -\gamma_{S_1} \\ -1 & -1 & 0 \end{pmatrix} \begin{pmatrix} a_1[k] \\ a_2[k] \\ a_3[k] \end{pmatrix}$$

As we can notice, in order to obtain a reflected wave, we do not need the relative incident wave. There is always a zero in the correspondent scattering matrix. For example, in the last equation we have

$$b_3[k] = (-1) \cdot a_1[k] + (-1) \cdot a_2[k] + 0 \cdot a_3[k]$$

That is what, practically speaking, adaptation means.

### Local non-linear scattering stage at the root

Anyway, at the end we bump into the non-linear element, the root. Here we can do nothing but applying the entire formula

$$a_3[k] = b_3[k] + 2Z_3I_s - 2\eta V_{th}W \left( \frac{Z_3I_s}{\eta V_{th}} e^{\frac{Z_3I_s+b_3[k]}{\eta V_{th}}} \right)$$

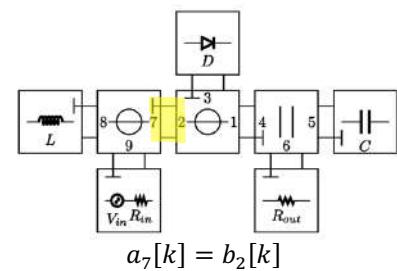
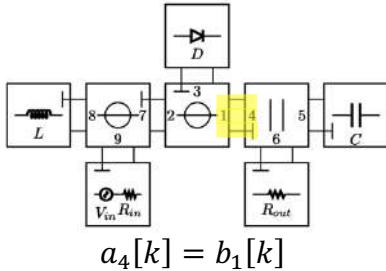
By computing the formula, we get the reflected wave by the diode  $a_3$  that we are going to use in the next step.

### Backward scan

We go back now to the leaves following in the same way as before but backwards. Therefore, we start now from the second layer of adaptors computing the waves reflecting by them toward the linear elements.

$$\begin{aligned} b_1[k] &= \gamma_{S_1}a_1[k] + (1 - \gamma_{S_1})a_2[k] + (1 - \gamma_{S_1})a_3[k] & \begin{pmatrix} b_1[k] \\ b_2[k] \\ b_3[k] \end{pmatrix} &= \begin{pmatrix} \gamma_{S_1} & (\gamma_{S_1} - 1) & (\gamma_{S_1} - 1) \\ -\gamma_{S_1} & (1 - \gamma_{S_1}) & -\gamma_{S_1} \\ -1 & -1 & 0 \end{pmatrix} \begin{pmatrix} a_1[k] \\ a_2[k] \\ a_3[k] \end{pmatrix} \\ b_2[k] &= -\gamma_{S_1}a_1[k] + (1 - \gamma_{S_1})a_2[k] - \gamma_{S_1}a_3[k] & \begin{pmatrix} b_1[k] \\ b_2[k] \\ b_3[k] \end{pmatrix} &= \begin{pmatrix} \gamma_{S_1} & (\gamma_{S_1} - 1) & (\gamma_{S_1} - 1) \\ -\gamma_{S_1} & (1 - \gamma_{S_1}) & -\gamma_{S_1} \\ -1 & -1 & 0 \end{pmatrix} \begin{pmatrix} a_1[k] \\ a_2[k] \\ a_3[k] \end{pmatrix} \end{aligned}$$

Ready to compute the waves reflected from the first layer of adaptors toward linear elements (so the waves incident to linear elements).



$$\begin{aligned} b_5[k] &= a_4[k] - \gamma_{P_1}a_5[k] + \gamma_{P_1}a_6[k] & \begin{pmatrix} b_4[k] \\ b_5[k] \\ b_6[k] \end{pmatrix} &= \begin{pmatrix} 0 & (1 - \gamma_{P_1}) & \gamma_{P_1} \\ 1 & \gamma_{P_1} & \gamma_{P_1} \\ 1 & (1 - \gamma_{P_1}) & (\gamma_{P_1} - 1) \end{pmatrix} \begin{pmatrix} a_4[k] \\ a_5[k] \\ a_6[k] \end{pmatrix} \\ b_6[k] &= a_4[k] + (1 - \gamma_{P_1})a_5[k] + (\gamma_{P_1} - 1)a_6[k] & \begin{pmatrix} b_4[k] \\ b_5[k] \\ b_6[k] \end{pmatrix} &= \begin{pmatrix} 0 & (1 - \gamma_{P_1}) & \gamma_{P_1} \\ 1 & \gamma_{P_1} & \gamma_{P_1} \\ 1 & (1 - \gamma_{P_1}) & (\gamma_{P_1} - 1) \end{pmatrix} \begin{pmatrix} a_4[k] \\ a_5[k] \\ a_6[k] \end{pmatrix} \end{aligned}$$

$$\begin{aligned} b_8[k] &= -\gamma_{S_2}a_7[k] + (1 - \gamma_{S_2})a_8[k] - \gamma_{S_2}a_9[k] & \begin{pmatrix} b_7[k] \\ b_8[k] \\ b_9[k] \end{pmatrix} &= \begin{pmatrix} 0 & -1 & -1 \\ -\gamma_{S_2} & (1 - \gamma_{S_2}) & -\gamma_{S_2} \\ (\gamma_{S_2} - 1) & (\gamma_{S_1} - 1) & \gamma_{S_2} \end{pmatrix} \begin{pmatrix} a_7[k] \\ a_8[k] \\ a_9[k] \end{pmatrix} \\ b_9[k] &= (\gamma_{S_2} - 1)a_7[k] + (\gamma_{S_1} - 1)a_8[k] + \gamma_{S_2}a_9[k] & \begin{pmatrix} b_7[k] \\ b_8[k] \\ b_9[k] \end{pmatrix} &= \begin{pmatrix} 0 & -1 & -1 \\ -\gamma_{S_2} & (1 - \gamma_{S_2}) & -\gamma_{S_2} \\ (\gamma_{S_2} - 1) & (\gamma_{S_1} - 1) & \gamma_{S_2} \end{pmatrix} \begin{pmatrix} a_7[k] \\ a_8[k] \\ a_9[k] \end{pmatrix} \end{aligned}$$

At the end, we iterate the algorithm since we have the instantaneous dependence of a port on the others.

## MATLAB implementation

Let us see how to implement it

```

%% Discrete-time Implementantion of the Envelope Follower based on WDFs
% Alberto Bernardini - 10/05/2020
clear
close all
clc

%% Input Audio Signal
[Vin,Fs]=audioread('ManoucheGuitarCompr.wav');
%% Sampling Period
Ts=1/Fs;
%% Number of Samples
Nsamp=length(Vin);
%% Simulated time
tstop=Nsamp*Ts;
%% Circuit Parameters
C=10*10^(-6);
L=10^(-9);
Rout= 10*10^3;
Rin = 1;

%% WDF setting of free parameters (adaptation conditions)
Z5=Ts/(2*C);
Z8=(2*L)/Ts;
Z6=Rout;
Z9=Rin;
% Make Ports of Adaptors Reflection-Free
Z4=(Z5*Z6)/(Z5+Z6);
Z1=Z4;
Z7=Z8+Z9;
Z2=Z7;
Z3=Z1+Z2;

%% Computation of Scattering Matrices
gammaSer1 = Z2/(Z1+Z2);
Sser1 = [gammaSer1, (gammaSer1-1), (gammaSer1-1);
          -gammaSer1, (1-gammaSer1), -gammaSer1 ;
          -1 , -1 , 0 ;];
gammaSer2 = Z8/(Z8+Z9);
Sser2 = [ 0 , -1 , -1 ;
          -gammaSer2 , (1-gammaSer2), -gammaSer2 ;
          (gammaSer2-1) , (gammaSer2-1), gammaSer2 ];
gammaPar1 = Z5/(Z5+Z6);
Spar1 = [ 0 , (1-gammaPar1), gammaPar1 ;
          1 , -gammaPar1 , gammaPar1 ;
          1 , (1-gammaPar1),(gammaPar1-1);];
%%% Alternatively (using fundamental loop matrix or fundamental cut-set
matrix)
% Bser = [1,1,1];
% Qpar = [1,1,1];
% Zser1= diag([Z1,Z2,Z3]);
% Zser2= diag([Z7,Z8,Z9]);
% Zpar1= diag([Z4,Z5,Z6]);
% Sser1= eye(3) - 2*Zser1*Bser'*inv(Bser*Zser1*Bser')*Bser;
% Sser2= eye(3) - 2*Zser2*Bser'*inv(Bser*Zser2*Bser')*Bser;
% Spar1= 2*Qpar'*inv(Qpar*inv(Zpar1)*Qpar')*Qpar*inv(Zpar1) - eye(3);

%% Initialization of Waves
b5=0;

```

```

b8=0;
a6=0;
% Numerically infer Initial Conditions for b5 and b8 (Variables of Dynamic
Elements)
% In case initial conditions are zero and the first sample of the Input
% signal is zero, this loop is not necessary (everything can be initialized to
zero)
ii=0;
while (ii<50000)
    ii=ii+1;

    % Constant Input Signal
    a9=Vin(1);
    % Manage Dynamic Elements
    a5=b5;
    a8=-b8;
    % Forward Scan
    b4 = Spar1(1,:)*[0;a5;a6];
    b7 = Sser2(1,:)*[0;a8;a9];
    a1=b4;
    a2=b7;
    b3= Sser1(3,:)*[a1;a2;0];
    % Local Nonlinear Scattering
    a3 = ExpDiodeLambert(b3,Z3);
    % Backward Scan
    b1 = Sser1(1,:)*[a1;a2;a3];
    b2 = Sser1(2,:)*[a1;a2;a3];
    a4=b1;
    a7=b2;
    b5 = Spar1(2,:)*[a4;a5;a6];
    b8 = Sser2(2,:)*[a7;a8;a9];
end

%% Initialize Output Signals
Vout=zeros(size(Vin));

%% Simulation Loop
ii=0;
while (ii<Nsamp)
    ii=ii+1;

    %% Manage Input Signal
    a9=Vin(ii);
    %% Manage Dynamic Elements
    a5=b5;
    a8=-b8;
    %% Forward Scan
    b4 = Spar1(1,:)*[0;a5;a6];
    b7 = Sser2(1,:)*[0;a8;a9];
    a1=b4;
    a2=b7;
    b3= Sser1(3,:)*[a1;a2;0];
    %% Local Nonlinear Scattering
    a3 = ExpDiodeLambert(b3,Z3);
    %% Backward Scan
    b1 = Sser1(1,:)*[a1;a2;a3];
    b2 = Sser1(2,:)*[a1;a2;a3];
    a4=b1;
    a7=b2;
    b5 = Spar1(2,:)*[a4;a5;a6];
    b6 = Spar1(3,:)*[a4;a5;a6];

```

```

b8 = Sser2(2,:)*[a7;a8;a9];
b9 = Sser2(3,:)*[a7;a8;a9];
%% Read Output
Vout(ii)=(a6+b6)/2;
end

%% LTSpice Files
load('Vout_LTspice.txt');
timeLTSpice = Vout_LTspice(:,1);
VoutLTSpice = Vout_LTspice(:,2);

figure
set(gcf, 'Color', 'w');
plot(timeLTSpice,VoutLTSpice,'r','Linewidth',2); hold on;
plot(Ts*[1:Nsamp],Vout,'b--','Linewidth',1); grid on;
xlabel('time [seconds]', 'Fontsize',16, 'interpreter','latex');
ylabel('$V_{\mathit{out}}$ [Volt]', 'Fontsize',16, 'interpreter','latex');
xlim([0,8]);
legend('LTspice','WDF', 'Fontsize',16, 'interpreter','latex');
set(gca, 'FontSize',15);

```

Where the script of the function ExpDiodeLambert() is

```

function [b] = ExpDiodeLambert(a,Z)
    %% Diode Parameters
    Is=50*10^(-9);
    eta=1;
    Vth=26*10^(-3);%25.8563*10^(-3);
    beta=eta*Vth;

    %% Reflected Wave Computation
    arg = log((Z*Is)/beta) + (a+Z*Is)/beta;
    expArg=exp(arg);
    if expArg==inf
        %% Omega Wright
        apprLevel=4;
        b = a + 2*Z*Is - 2*beta* OmegaWrightDangelo(arg,apprLevel);
    else
        %% Lambert
        b = a + 2*Z*Is - 2*beta* Lambert_W_Fritsch(expArg);
    end
end

%% Lambert_W_Fritsch
function [ w ] = Lambert_W_Fritsch( x )
% This function evaluates the upper branch of the Lambert-W function for
% real input x. Function is not optimized for vectorial processing.

% Values
%
% x - Input Range: [0,inf)
% w - W(x)

% Function written by F. Esqueda 2/10/17 based on implementation presented
% by Darko Veberic - "Lambert W Function for Applications in Physics"
% Available at https://arxiv.org/pdf/1209.0735.pdf

%% Compute Lambert-W

```

```
% Error threshold
thresh = 10e-12;

if x==0
    w=0;
else
    % Initial guess
    if x<0.14546954290661823

        num = 1 + 5.931375839364438*x + 11.39220550532913*x^2 +
7.33888339911111*x^3 + 0.653449016991959*x^4;
        den = 1 + 6.931373689597704*x + 16.82349461388016*x^2 +
16.43072324143226*x^3 + 5.115235195211697*x^4;
        w = x * num / den;

    elseif x<8.706658967856612

        num = 1 + 2.4450530707265568*x + 1.3436642259582265*x^2 +
0.14844005539759195*x^3 + 0.0008047501729130*x^4;
        den = 1 + 3.4447089864860025*x + 3.2924898573719523*x^2 +
0.9164600188031222*x^3 + 0.05306864044833221*x^4;
        w = x * num / den;

    else

        a = log(x);
        b = log(a);
        ia = 1/a;
        w = a - b + (b*ia) * 0.5*b*(b - 2)*(ia*ia) + (1/6)*(2*b*b - 9*b +
6)*(ia*ia*ia);

    end

    % Compute Lambert-W (limited to 20 iterations)
    for m=1:20

        w1 = w + 1;
        z = log(x) - log(w) - w;
        q = 2*w1*(w1 + (2/3)*z);
        e = (z/w1)*((q - z)/(q - 2*z));
        w = w * (1 + e);

        if abs(e)<thresh
            break;
        end
    end
end

%% OmegaWrightDangelo
function [y] = OmegaWrightDangelo(x,apprLevel)

switch apprLevel
    case 1
        y = omega1(x);
    case 2
        y = omega2(x);
    case 3
        y = omega3(x);
    case 4
        y = omega4(x);
end
```

```

    otherwise
        y = wrightOmegaq(x);
    end
end

function y = omega1(x)
    y = max(0, x);
end

function y = omega2(x)
    x1 = -3.684303659906469;
    x2 = 1.972967391708859;
    a = 9.451797158780131e-3;
    b = 1.126446405111627e-1;
    c = 4.451353886588814e-1;
    d = 5.836596684310648e-1;
    y = x;
    y(x<x2) = d + x(x<x2) .* (c + x(x<x2) .* (b + x(x<x2) .* a));
    y(x<x1) = 0;
end

function y = omega3(x)
    x1 = -3.341459552768620;
    x2 = 8;
    a = -1.314293149877800e-3;
    b = 4.775931364975583e-2;
    c = 3.631952663804445e-1;
    d = 6.313183464296682e-1;
    y = x - logApprox(x);
    y(x<x2) = d + x(x<x2) .* (c + x(x<x2) .* (b + x(x<x2) .* a));
    y(x<x1) = 0;
end

function y = omega4(x)
    y = omega3(x);
    y = y - (y - expApprox(x - y)) ./ (y + 1);
end

% the following functions simulate the log/exp approximations described in the
paper
function y = log2Approx(x)
    e = floor(log2(x));
    d = x ./ (2 .^ e);
    y = e - 2.213475204444817 + d .* (3.148297929334117 + d .* (-
1.098865286222744 + d .* 0.1640425613334452));
end

function y = logApprox(x)
    y = 0.693147180559945 * log2Approx(x);
end

function y = pow2Approx(x)
    l = floor(x);
    f = x - l;
    y = (2 .^ l) .* (1.0 + f .* (0.6931471805599453 + f .* (0.2274112777602189 +
f .* 0.07944154167983575)));
end

function y = expApprox(x)
    y = pow2Approx(1.442695040888963 * x);
end

```

```

%% Source: https://it.mathworks.com/help/symbolic/wrightomega.html
function W=wrightOmegaq(Z)
%WRIGHTOMEGAQ Wright omega function, a solution of the equation W+LOG(W) = Z.
%   W = WRIGHTOMEGAQ(Z) performs floating point evaluation of the Wright omega
%   function. Z is an array and may be complex. If Z is an array of symbolic
%   values, it is converted to double-precision for computation and then
recast
%   as symbolic.
%
% Example:
%   % Plot magnitude of the function surface over the complex plane
%   v = -6*pi:3*pi/25:6*pi; [x,y] = meshgrid(v);
%   w = wrightOmegaq(x+y*1i);
%
% figure('Renderer','zbuffer'); surf(v,v,abs(w)); colormap(hot(256));
% shading flat; axis square; xlabel('X'); ylabel('Y');
% zlabel('|omega(X+Yi)|'); title('Wright \omega Function')
%
% Note:
% Due to numerical precision and the nature of this equation, the
inverse
% of the Wright omega function, may not always return a value close (in
an
% absolute sense) to Z, e.g., Z <= -714.84989998498+Y*1i, -pi < Y <= pi.
%
% Class support for Z:
%   float: double, single
%   symbolic
%
% See also: WRIGHTOMEGA, LAMBERTW
% Based on:
%
% [1] Piers W. Lawrence, Robert M. Corless, and David J. Jeffrey, "Algorithm
% 917: Complex Double-Precision Evaluation of the Wright omega Function,"  

ACM
% Transactions on Mathematical Software, Vol. 38, No. 3, Article 20, pp. 1-
17,
% Apr. 2012. http://dx.doi.org/10.1145/2168773.2168779
%
% [2] Robert M. Corless and David J. Jeffrey, "The Wright omega Function,"  

In:
% Artificial Intelligence, Automated Reasoning, and Symbolic Computation,
% Joint International Conferences, AISC 2002 and Calculemus 2002, Marseille,
% France, July 2002, (Jacques Calmet, Belaid Benhamou, Olga Caprotti,  

Laurent
% Henocque, and Volker Sorge, Eds.), Berlin: Springer-Verlag, pp. 76-89,
2002.
% http://orcca.on.ca/TechReports/2000/TR-00-12.html
%
% Numbers in parentheses below refer to equations in Lawrence, et al.
2012.
% The inverse Wright omega function is defined as (Corless & Jeffrey,
2002):
%   Z = W+LOG(W)-2*pi*1i,      -Inf < REAL(W) < -1, IMAG(W) = 0
%   Z = -1+/-pi*1i,           W = -1
%   Z = W+LOG(W),            otherwise
% The solution of W+LOG(W) = Z is given by (Corless & Jeffrey, 2002):
%   W = WRIGHTOMEGA(Z),          Z ~ t+/-pi*1i, t <= -1
%   W = WRIGHTOMEGA(Z),WRIGHTOMEGA(Z-2*pi*1i),  Z = t+pi*1i, t <= -1
%   W = NaN,                      Z = t-pi*1i, t <= -1

```

```
% WRIGHTOMEGAQ is up three to four orders of magnitude faster than
WRIGHTOMEGA
% for double-precision arrays. Additionally, it has much less numeric error,
% properly evaluates values greater than 2^28, supports single-precision
% evaluation, and handles NaN inputs.
% Andrew D. Horchler, horchler @ gmail . com, Created 7-12-12
% Revision: 1.0, 3-12-13
% Convert symbolic input, converted back at end
isSym = isa(Z, 'sym') || ischar(Z);
if isSym
    try
        Z = double(sym(Z));
    catch ME
        if strcmp(ME.identifier, 'symbolic:sym:double:cantconvert')
            error('SHCTools:wrightOmegaq:InvalidSymbolicZ',...
                ['Symbolic input Z must contain only numeric values and no
'...
                'expressions containing variables.'])
        else
            rethrow(ME);
        end
    end
elseif ~isfloat(Z)
    error('SHCTools:wrightOmegaq:InvalidZ',...
        'Input Z must be an array of floating point or symbolic values.');
end
% Support for single precision: single(pi) ~= pi
dataType = class(Z);
PI = cast(pi,dataType);
PI2 = cast(pi/2,dataType);
EXP1 = cast(exp(1),dataType);
EXP2 = cast(exp(2),dataType);
LN2 = cast(log(2),dataType);
OMEGA = cast(0.5671432904097838,dataType);
ONE_THIRD = cast(1/3,dataType);
tol = eps(dataType);
X = real(Z(:));
Y = imag(Z(:));
if isempty(Z) || all(isnan(Z(:)))
    W = Z;
elseif isscalar(Z)
    % Special values
    if Z > 2^59
        W = Z; % W self-saturates: X > 2^59 (abs(Y) > 2^54
    too)
        elseif Z == 0
            W(1) = OMEGA; % Omega constant
        elseif Z == 1
            W(1) = 1;
        elseif Z == 1+EXP1
            W(1) = EXP1;
        elseif isreal(Z) || Y == 0
            if Z < log(eps(realmin(dataType)))-LN2
                W(1) = 0; % Z -> -
            Inf
            else
                % W(1) used in order retain datatype
                if Z <= -2
                    % Region 3: series about -Inf
                    x = exp(X);
                    W(1) = x*(1-x*(1-x*(36-x*(64-125*x))/24)); % (24)
                end
            end
        end
    end
else
    % W(1) used in order retain datatype
    if Z <= -2
        % Region 3: series about -Inf
        x = exp(X);
        W(1) = x*(1-x*(1-x*(36-x*(64-125*x))/24)); % (24)
    end
end
```

```

%fprintf(1,'W: %.20f\n',W);
    % Series is exact, X < -exp(2)
    if X < -EXP2
        return;
    end
elseif Z > PI+1
    % Region 7: log series about Z = Inf
    x = log(X);
    lzi = x/X;
    W(1) = X-x+lzi*(1+lzi*(0.5*x-1 ...
        +lzi*((ONE_THIRD*x-1.5)*x+1))); % (25)
else
    % Region 4: series about Z = 1
    x = X-1;
    W(1) = 1+x*(1/2+x*(1/16-x*(1/192+x*(1/3072 ...
        -(13/61440)*x)))); % (29)
end

% Residual
r = X-(W+log(W)); % (14)

%fprintf(1,'W: %.20f\n',W);
%fprintf(1,'r: %.20f\n',r);
if abs(r) > tol
    % FSC-type iteration, N = 3, (Fritsch, Shafer, & Crowley,
1973)
    w1 = 1+W;
    w2 = w1+2*ONE_THIRD*r;
    W = W*(1+r*(w1*w2-0.5*r)/(w1*(w1*w2-r))); % (15)
    % Test residual
    r = X-(W+log(W));
    %sprintf('Zr: %.20f\n',X) % (14)

%fprintf(1,'W: %.20f\n',W);
%fprintf(1,'ln W: %.20f\n',log(W));
%fprintf(1,'r: %.20f\n',r);
%sprintf('W+ln W: %.20f\n',W + log(W))
%sprintf('r: %.20f',r)
    % Second iterative improvement via FSC method, if needed
    if abs(r) > tol
        w1 = 1+W;
        w2 = w1+2*ONE_THIRD*r;
        W = W*(1+r*(w1*w2-0.5*r)/(w1*(w1*w2-r))); % (15)
    end
end
end
else
    % Special complex values
    if Z < log(eps(realmin(dataType)))-LN2 && Y > -PI && Y <= PI
        W(1) = 0; % Z -> -
Inf
    elseif Z == -1+PI*i || Z == -1-PI*i
        W(1) = -1;
    elseif Z == log(ONE_THIRD)-ONE_THIRD+PI*i
        W(1) = -ONE_THIRD;
    elseif Z == LN2-2-PI*i
        W(1) = -2;
    elseif Z == (1+PI2)*i
        W(1) = 1i;
    else
        xgtm2 = (X > -2);
    end
end

```

```

% W(1) used in order retain datatype
if X <= -1 && abs(Y) == PI
    % Upper and lower lines of discontinuity
    if Y == PI
        % Region 3: series about -Inf
        x = exp(X);
        W(1) = ((((-125*x-64)*x-36)*x/24-1)*x-1)*x;           % (24)

        % Series is exact, X < -exp(2)
        if X < -EXP2
            return;
        end
    elseif xgtm2
        % Regions 1 and 2: near Z = -1+pi*i and Z = -1-pi*i
        x = sign(Y)*sqrt(-2*(X+1));                                % (20),
22)
        W(1) = -1+x*(1-x*(1440-x*(120+x*(16+x)))*(1/4320));      %

(21, 23)
        else
            % Region 6: negative log series about Z = Z+pi*i
            x = log(-X);
            lzi = x/X;
            W(1) = X-x+lzi*(1+lzi*(0.5*x-1+lzi*((ONE_THIRD*x-
1.5)*x+1)));          % (28)
        end
%fprintf(1,'W: %.20f\n',W);
        % Regularization: adjust Z and flip sign of W
        Z = X;
        s = -1;
    else
        xlteq1 = (X <= 1);
        r12x = (xgtm2 && xlteq1);

        if ~xgtm2 && Y > -PI && Y < PI
            % Region 3: series about -Inf, within lines of
discontinuity
            x = exp(Z);
%fprintf(1,'x: %.30f%.30f\n',real(x),imag(x));
            W(1) = x*(1-x*(1-x*(36-x*(64-125*x))*(1/24)));      % (24)
%fprintf(1,'W: %.30f%.30f\n',real(W),imag(W));
            % Series is exact, X < -exp(2)
            if X < -EXP2
                return;
            end
        elseif r12x && Y > PI2 && Y < 3*PI2
            % Region 1: near Z = -1+pi*i, but not line of
discontinuity
            x = conj(sqrt(2*conj(Z+1-PI*i)));                      % (20)
%fprintf(1,'x: %.30f%.30f\n',real(x),imag(x));
            %W(1) = -1+x*1i+x**x*(1440+16*x**x ...
            %      +(x**x-120)*x*1i)*(1/4320);
            %W(1) = -1+x*1i+(1440*x*x+16*x*x*x*x ...
            %      +(x**x-120)*x*x*x*x1i)/4320;
            %W(1) = -1+x*1i+(1440*(x**x)-
120*(x*(x*(x*1i)))+16*(x*(x*(x*x))) ...
            %      +x*(x*(x*(x*(x*1i)))))/4320;
            W(1) = -1+x*(1i+x*(1440-x*(120*1i ...
            -x*(16+x*1i)))*(1/4320));                                % (21)
%fprintf(1,'W: %.30f%.30f\n',real(W),imag(W));
        elseif r12x && Y > -3*PI2 && Y < -PI2

```

```

        % Region 2: near Z = -1-pi*1i, but not line of
discontinuity
        x = conj(sqrt(2*conj(Z+1+PI*1i)));                                % (22)
        W(1) = -1-x*(1i-x*(1440+x*(120*1i ...
            +x*(16-x*1i)))*(1/4320));                                         % (23)
        %W(1) = -1-x*1i+x*x*(1440+16*x*x ...
        %      +(120-x*x)*x*x*1i)/4320;
        elseif r12x && abs(Y) <= PI2 || ~xlteq1 && (X-1)^2+Y^2 <= PI^2
        % Region 4: series about Z = 1
        x = Z-1;
        W(1) = 1+x*(1/2+x*(1/16-x*(1/192+x*(1/3072 ...
            -(13/61440)*x)));                                              % (29)
        elseif ~xgtm2 && Y > PI && Y-PI <= -0.75*(X+1)
        % Region 5: negative log series about t = Z-pi*1i
        t = Z-PI*1i;                                                       %
(26)      x = log(-t);
        lzi = x/t;
        W(1) = t-x+lzi*(1+lzi*(0.5*x-1+lzi*((ONE_THIRD*x-
1.5)*x+1)));                                                 % (28)
        elseif ~xgtm2 && Y < -PI && Y+PI >= 0.75*(X+1)
        % Region 6: negative log series about t = Z+pi*1i
        t = Z+PI*1i;                                                       % (26)
        x = log(-t);
        lzi = x/t;
        W(1) = t-x+lzi*(1+lzi*(0.5*x-1+lzi*((ONE_THIRD*x-
1.5)*x+1)));                                                 % (28)
        else
            % Region 7: log series about Z = Inf
            x = log(Z);
%fprintf(1,'x: %.30f%.30f\n',real(x),imag(x));
            lzi = x/Z;
%fprintf(1,'lzi: %.30f%.30f\n',real(lzi),imag(lzi));
            W(1) = Z-x+lzi*(1+lzi*(0.5*x-1+lzi*((ONE_THIRD*x-
1.5)*x+1)));                                                 % (25)
%fprintf(1,'W: %.30f%.30f\n',real(W),imag(W));
            end

            % Check for regularization: adjust Z and flip sign of W
            if X <= -0.99 && abs(Y-PI) <= 0.01
                Z = Z-PI*1i;                                                       %
(26)          s = -1;
            elseif X <= -0.99 && abs(Y+PI) <= 0.01
                Z = Z+PI*1i;                                                       % (26)
                s = -1;
            else
                s = 1;
            end
            end

            % Residual (can be zero)
            r = Z-(W+log(s*W));                                              % (14)
%fprintf(1,'r: %.20f\n',r);
            if abs(r) > tol
                % FSC-type iteration, N = 3, (Fritsch, Shafer, & Crowley,
1973)
                %W = W*(1+(r/(1+W))*((1+W)*(1+W+2*ONE_THIRD*r)...
                %      -0.5*r)/((1+W)*(1+W+2*ONE_THIRD*r)-r));                 %
(15)          w1 = 1+W;

```

```

w2 = w1+2*ONE_THIRD*r;
W = W*(1+r*(w1*w2-0.5*r)/(w1*(w1*w2-r)));
%W = W*(1+(r/(1+W))*(1+W)*(1+W+2*ONE_THIRD*r)...
%      -0.5*r)/((1+W)*(1+W+2*ONE_THIRD*r)-r)); % (14)
(15)   % Test residual
r = Z-(W+log(s*W));

% Second iterative improvement via FSC method, if needed
if abs(r) > tol
    w1 = 1+W;
    w2 = w1+2*ONE_THIRD*r;
    W = W*(1+r*(w1*w2-0.5*r)/(w1*(w1*w2-r)));
    %W = W*(1+(r/(1+W))*(1+W)*(1+W+2*ONE_THIRD*r)...
    %      -0.5*r)/((1+W)*(1+W+2*ONE_THIRD*r)-r)); %
(15)   end
        end
    end
else
    isRealZ = (isreal(Z) || all(Y == 0));
    if isRealZ
        W = NaN(size(Z), dataType);
    else
        W = complex(NaN(size(Z), dataType));
    end

    % Special values
    W(Z == 0) = 0.5671432904097838; % Omega
constant
    W(Z == 1) = 1;
    W(Z == 1+exp(1)) = exp(1);
    W(Z > 2^59) = Z(Z > 2^59); % W self-saturates: X > 2^59 (abs(Y) >
2^54 too)

    if isRealZ
        MinZ = (Z < log(eps(realmin(dataType)))-log(2));
        W(MinZ) = 0; % Z -> -
Inf
        else
            MinZ = (Z(:) < log(eps(realmin(dataType)))-log(2) & Y > -PI & Y <=
PI);
            W(MinZ) = 0; % Z -> -
Inf
        W(Z == -1+PI*1i | Z == -1-PI*1i) = -1;
        W(Z == log(1/3)-1/3+PI*1i) = -1/3;
        W(Z == log(2)-2-PI*1i) = -2;
        W(Z == (1+PI/2)*1i) = 1i;
    end

    i = (isnan(W(:)) & ~isnan(Z(:)));
    if any(i)
        if isRealZ
            % Region 3: series about -Inf
            j = (i & X <= -2);
            if any(j)
                x = exp(X(j));
                W(j) = x.*((1-x.*((1-x.*((36-x.*((64-125*x))/24)))))); % (24)
            end
        end
    end
end

```

```

        if all(~i | X < -7.38905609893065)
            return;
        end
    end

    % Region 7: log series about Z = Inf
    j = (~j & X > pi+1);
    if any(j)
        t = X(j);
        x = log(t);
        lzi = x./t;
        W(j) = t-x+lzi.*((1+lzi.*((x/2-1+lzi.*((x/3-3/2).*x+1)))));      %
    end

    % Region 4: series about Z = 1
    j = (~j & X > -2);
    if any(j)
        x = X(j)-1;
        W(j) = 1+x.*(1/2+x.*(1/16-x.*(1/192+x.*(1/3072 ...
            -(13/61440)*x))));          % (29)
    end

    % No regularization
    Z = real(Z);
    s = 1;
else
    ypi = (Y == PI);
    ympi = (Y == -PI);
    xgtm2 = (X > -2);
    xlteq1 = (X <= 1);
    r12x = (i & xgtm2 & xlteq1);

    % Region 3b: series about -Inf, upper line of discontinuity
    c = (i & ~xgtm2 & ypi);
    if any(c)
        x = exp(X(c));
        W(c) = ((((-125*x-64).*x-36).*x/24-1).*x-1).*x;          % (24)

        % Series is exact, X < -exp(2)
        i = (i & X >= -7.38905609893065);
        if all(~i)
            return;
        end
    end
c7 = c;

    % Region 3: series about -Inf, Z between lines of discontinuity
    c = (i & ~xgtm2 & Y > -PI & Y < PI);
    if any(c)
        x = exp(Z(c));
        W(c) = x.*(1-x.*(1-x.*(36+x.*(64+125*x))/24));          % (24)

        % Series is exact, X < -exp(2)
        if all(~i | X < -7.38905609893065)
            return;
        end
    end
c7 = (c7 | c);

    % Regions 1b and 2b: near Z = -1+/-pi*i, lines of discontinuity

```

```

c = (i & xgtm2 & X <= -1 & (ypi | ympi));
if any(c)
    x = sign(Y(c)).*sqrt(-2*(X(c)+1)); % (20, 22)
    W(c) = -1+x.*(1-x.*(1440-x.*(120+x.*(16+x)))/4320); % (21,
23)
end
c7 = (c7 | c);

% Region 1: near Z = -1+pi*1i, but not upper line of discontinuity
c = (~ypi & r12x & Y > PI/2 & Y < 3*PI/2);
if any(c)
    x = conj(sqrt(2*conj(Z(c)+1-PI*1i))); % (20)
    W(c) = -1+x*1i+(1440*x.^2-120*x.^3*1i+16*x.^4 ...
        +x.^5*1i)/4320; % (21)
end
c7 = (c7 | c);

% Region 2: near Z = -1-pi*1i, but not lower line of discontinuity
c = (~ypi & r12x & Y > -3*PI/2 & Y < -PI/2);
if any(c)
    x = conj(sqrt(2*conj(Z(c)+1+PI*1i))); % (22)
    W(c) = -1-x*1i+(1440*x.^2+120*x.^3*1i+16*x.^4 ...
        -x.^5*1i)/4320; % (23)
end
c7 = (c7 | c);

% Region 4: series about Z = 1
c = (r12x & abs(Y) <= PI/2 | ~xlteq1 & (X-1).^2+Y.^2 <= PI^2);
if any(c)
    x = Z(c)-1;
    W(c) = 1+x.*((1/2+x.*((1/16-x.*((1/192+x.*((1/3072 ...
        -(13/61440)*x)))))); % (29)
end
c7 = (c7 | c);

% Region 5: negative log series about t = Z-pi*1i
c = (i & ~xgtm2 & Y > PI & Y-PI <= -(3/4)*(X+1));
if any(c)
    t = Z(c)-PI*1i; % (26)
    x = log(-t);
    lzi = x./t;
    W(c) = t-x+lzi.*((1+lzi.*((x/2-1+lzi.*((x/3-3/2).*x+1))))); %
(28)
end
c7 = (c7 | c);

% Region 6b: negative log series about t = Z+pi*1i
c = (i & ~xgtm2 & ympi);
if any(c)
    t = X(c); % (26)
    x = log(-t);
    lzi = x./t;
    W(c) = t-x+lzi.*((1+lzi.*((x/2-1+lzi.*((x/3-3/2).*x+1))))); %
(28)
end
c7 = (c7 | c);

% Region 6: negative log series about t = Z+pi*1i
c = (i & ~xgtm2 & Y < -PI & Y+PI >= (3/4)*(X+1));
if any(c)
    t = Z(c)+PI*1i; % (26)

```

```

x = log(-t);
lzi = x./t;
W(c) = t-x+lzi.*(1+lzi.*((x/2-1+lzi.*((x/3-3/2).*x+1)));      %
(28)
end
c7 = (c7 | c);

% Region 7: log series about Z = Inf
c = (i & ~ (c7 | c));
if any(c)
    t = Z(c);
    x = log(t);
    lzi = x./t;
    W(c) = t-x+lzi.*(1+lzi.*((x/2-1+lzi.*((x/3-3/2).*x+1)));      %
(25)
end

% Check for regularization: adjust Z and flip sign of W
s1 = (X <= -0.99 & abs(Y-PI) <= 1e-2);
Z(s1) = Z(s1)-PI*1i;                                         % (26)

s2 = (X <= -0.99 & abs(Y+PI) <= 1e-2);
Z(s2) = Z(s2)+PI*1i;                                         % (26)

s = ones(size(W));
s(s1 | s2) = -1;
end

% Residual (can be zero)
r = Z-(W+log(s.*W));                                         % (14)
r(MinZ) = 0;

% FSC-type iteration, N = 3, (Fritsch, Shafer, & Crowley, 1973)
rc = (abs(r(:)) > tol);
if any(rc)
    wr = W(rc);
    r = r(rc);

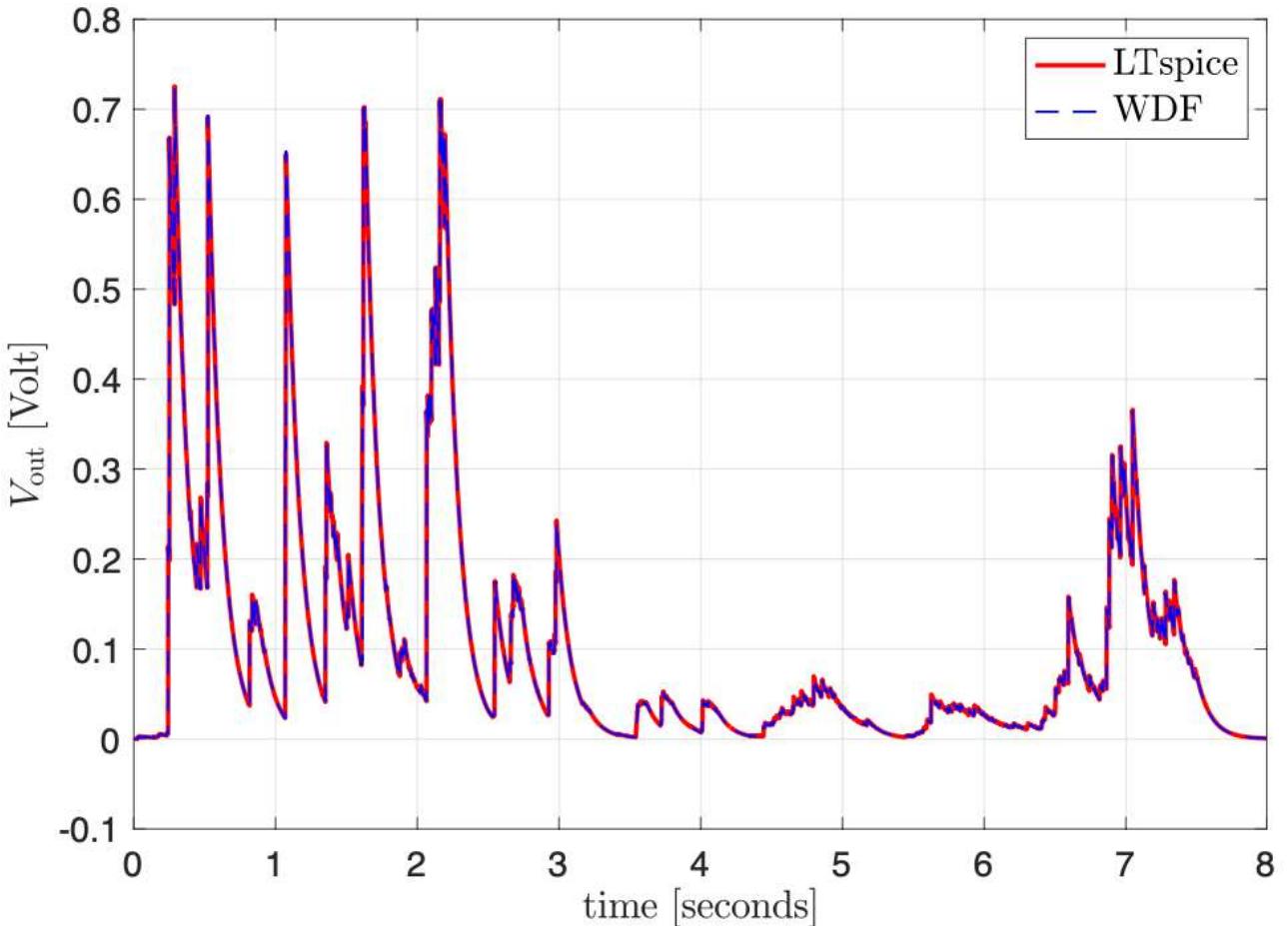
    W(rc) = wr.*((1+r./(1+wr)).*((1+wr).*(1+wr+(2/3)*r)...
        -r/2)./((1+wr).*(1+wr+(2/3)*r)-r));                  % (15)

    % Test residual
    r = Z-(W+log(s.*W));                                         % (14)
    r(MinZ) = 0;
    rc = (abs(r(:)) > tol);
    if any(rc)
        wr = W(rc);
        r = r(rc);

        % Second iterative improvement via FSC method, if needed
        W(rc) = wr.*((1+r./(1+wr)).*((1+wr).*(1+wr+(2/3)*r)...
            -r/2)./((1+wr).*(1+wr+(2/3)*r)-r));                % (15)
    end
end
end
end
% Reconvert symbolic input
if isSym
    W = sym(W, 'd');
end
end

```

This is the result we get compared to the LTspice simulation.



What we can say is that the two results perfectly match excluding the numerical difference due to the computational model itself. The difference between the two models lies in the speed with which we get the result hence in the efficiency of the model: the WDF method is incredibly faster with respect to the traditional simulation models. That is why is preferred in the modern simulation systems.

## References

- [1] A. Fettweis. Wave digital filters: Theory and practice. *Proc. IEEE*, 74(2):270–327, Feb. 1986.
- [2] A. Bernardini, P. Maffezzoni, and A. Sarti. Linear multistep discretization methods with variable step-size in nonlinear wave digital structures for virtual analog modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(11):1763–1776, Nov. 2019.
- [3] R. C. De Paiva, S. D’Angelo, J. Pakarinen, and V. Välimäki. Emulation of operational amplifiers and diodes in audio distortion circuits. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 59:688–692, Oct. 2012.
- [4] A. Bernardini, K. J. Werner, A. Sarti, and J. O. Smith III. Modeling nonlinear wave digital elements using the Lambert function. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 63(8):1231–1242, Aug. 2016.
- [5] A. Bernardini, K. J. Werner, J. O. Smith III, and A. Sarti. Generalized wave digital filter realizations of arbitrary reciprocal connection networks. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 66(2):694–707, Feb. 2019.
- [6] A. Sarti and G. De Sanctis. Systematic methods for the implementation of nonlinear wave-digital structures. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 56: 460–472, Feb. 2009.

## Spatial Sound with Headphones: Binaural Rendering

In today's lecture we will see how to create the perception of spatial sound with headphones.

### Immersive communication

When we use headphones, we want to provide a sense of immersivity for the listener. Our aim is to go beyond realism, to provide the natural binaural cues to enhance intelligibility.

As we can imagine, the applications are many:

- Teleconferencing, telepresence
- Place the listener in a performance venue, enable a new class of audio effects
- Gaming: awareness of sound-generating objects and surrounding acoustic space

### Challenges

In cheapest contemporary mobile phones sound is heard over only one earphone, or it is duplicated to two earphones (when available). If a single voice channel feeds both earphones, the listeners will perceive the voice near the centre of their head. Simply speaking, a pair of earphones enables binaural rendering. In order to create a sense of spatiality we can produce relevant auditory cues by changing balance and/or introducing ITD and ILD. Anyway, the effect is not exactly what we want:

- Shift apparent location to a different point on a line between the ears
- Sound remains inside the head (unnatural)

How can we overcome these limitations?

### Binaural recordings

One solution is provided by recording with two microphones placed in correspondence of the ears of a dummy head. In this way we are able to:

- Introduce realistic primary cues (ITD and ILD)
- Provide information on room reflections and reverberation

However, this type of recordings has some problems:

- front/back confusion: it is difficult to understand if a sound comes from front or back if it is on the medium plane. A real person, to solve this issue, just has to rotate the head.
- Failure in externalising sound sources in front or back.
- Sensitivity to actual size and shape of listener's head and outer ear: everyone one of us modifies the sound field differently since each ear and each head is unique and unrepeatable.
- It does not support the common experience of focusing attention by turning the head towards the sound source.

To solve this problem, we can exploit some dynamic cues. There are three main possible solutions:

- Use HRTF to filter the signals from the source to the listener's ears:

This technique requires having the HRTF and the isolated signals for each source in the scene. We use HRTF interpolation to account for head motion.

Basically, we obtain two transfer functions  $H_l(\omega)$  and  $H_r(\omega)$  that modify the signal to make it similar to what we perceive, respectively, with our left ear and our right ear. Then, we obtain the signal for both ears as  $Y_l(\omega)$  and  $Y_r(\omega)$

- Motion-tracked binaural (MTB)  
We measure the sound field sparsely in space around a real or virtual dummy head. We reconstruct the sound field for the desired location.
- In both cases (HRTF-based and MTB), dynamic cues are controlled by head motion.

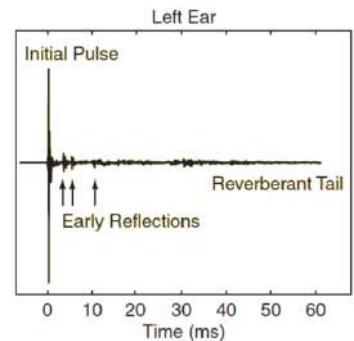
### HRTF-Based Binaural Rendering

HRTF produces a listening experience limited to anechoic environments. Since we are not considering reverberation, the sound becomes very unnatural (in the real world, we are always immersed in environment that generates reverberations). Furthermore, it impairs distance perception.

Therefore, to measure the impulse response in a room for each ear we have to

- include early reflections and reverberation
- include binaural cues and HRTF

As we can see from the image, in the binaural impulse response (BRIR) we can find both the early reflections and the reverberant tail. Both contain information on the environment where the sound is reproduced.



We can imagine that BRIRs are much longer than HRIRs since we have to take into account not only the initial pulse (included also in the HRIR), but also the early reflections and the reverberant tail. This leads to the issues of latency and computation time.

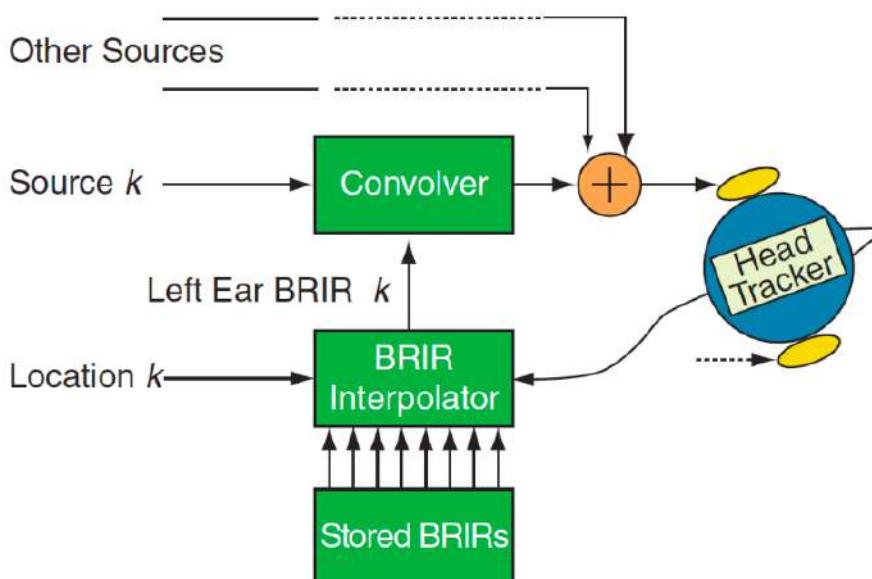
In order to solve the problem, we come up with the HRTF approach. It is used to provide virtual acoustic environments. Firstly, it was developed for:

- Computer games
- Military training systems

How does it work?

We suppose to have separate signals available for every source and to know the spatial locations of them. In this context, head tracker is used to determine location and orientation of the listener in the environment.

The image below shows the pipeline for the binaural room scanning system



The BRIR interpolator, based on the position of the user, exploit the BRIR database to create a filter that gives the sensation of spatiality. The filter is sent to the convolver that, in turns, filters the source signal. This is done for every  $k$  sources.

The development of a binaural scanning system is divided in two different phases:

- **Acquisition:**

Source signals are sent to loudspeakers positioned in a listening room. Then, BRIRs are measured from the loudspeakers to the microphones in a dummy head at ideal listening location. The measurement is performed for every few degrees of head rotation.

This generates a couple of issues: firstly, the dummy head is not exactly equal to the shape of our head; secondly, we have to choose a resolution for the sampling of the direction. The higher the resolution, the better the sensation of spatiality. Obviously, the higher the resolution, the higher the amount of data to be processed. What we need is a trade-off to make our system work.

- **Playback**

A signal coming from the head tracker controls an interpolator that combines adjacent BRIRs to produce left and right BRIRs that vary continuously with head motion.

Convolution of source signals and corresponding BRIRs are summed and fed to headphones.

If any of these passages is done in real time, we are able to give a high-quality spatial sound. To compensate the non-idealities of the shape of the head, we can exploit what we do naturally when this happens: to rotate our head in the direction from which the sound is coming.

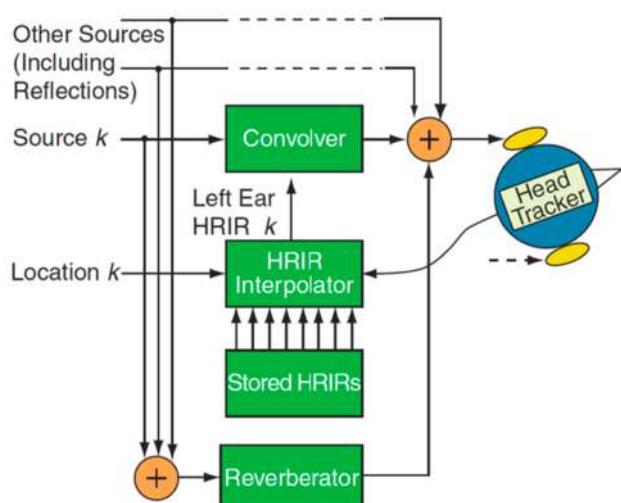
Let us now make some considerations on the implementations of such a system:

- Large numbers of long BRIRs must be measured and stored (144 ....?)
- The distortion due to interpolation must be negligible: it has to introduce no latencies and no artefacts.
- Transfer functions of headphones must be compensated: obviously also the headphones and earphones are characterised by a certain transfer functions that modifies the sound perceived.

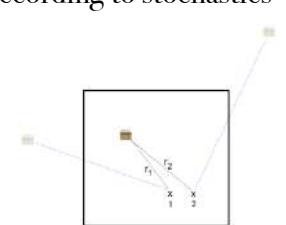
To give an idea of the computational cost of this kind of system let us see the following considerations.

Obviously, it depends on the dimension and reflectivity of the auditory scene. For example, considering a small room, the BRIRs length will be around 0.5 s. This means that we will have around 22000 samples that represent the function considering a sampling frequency equal to  $F_s = 44.1 \text{ kHz}$ . Convolution is always the most complex operation (with a complexity of  $O(n^2)$ ) in this kind of systems. In our specific case, it will require approximately 1 *Giga* of operations per second. Furthermore, the requirements are doubled for two ears, and they scale linearly with the number of desired sources. This is surely a problem as the head motion requires a fast adaptation of BRIRs.

## BRS with room model



As we said, we need to include the room model in our BRS. Since we want to reduce the complexity of our system, we modify the pipeline for acquisition. This time again, we store some information, but we discard every information that comes from the room. It will be handled by a separated line and modelled by a reverberator that generates the effect according to stochastic information. How can we recreate the sound field generated from the first reflection? With the same “specularity game” as always.

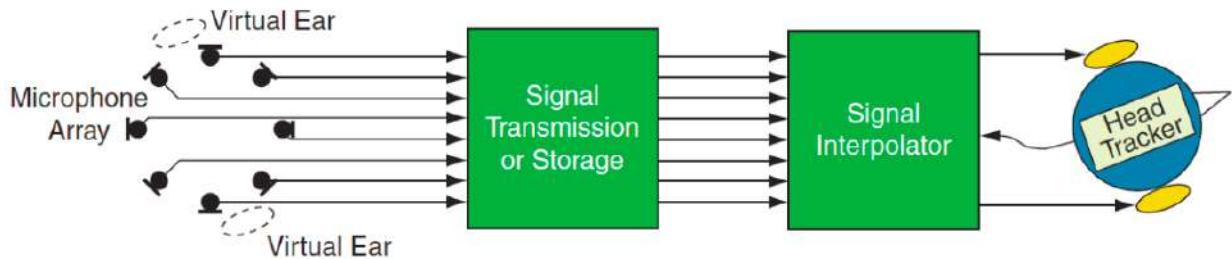


This type of representation is suitable to single-listener virtual environment: source locations are under control (the system is position-dependent).

On the other hand, it is not suited to capture natural sound. For this kind of sounds it is hard to obtain separate source signals, to determinate their position and, if that was not enough, it is computationally expansive to capture reflections and reverberations in a natural listening space.

In commercial systems, a two-stage approach is adopted to perform this technique: conventional recording practices are used to produce a surround sound mix, then HRIR-based processing is applied

### Motion-Tracked Binaural



For this type of system, we use a circular array system. This type of method is independent from the acoustic environment, we do not need to know the number of sources and their position. In this type of application, the microphones are mounted in a structure that resembles in some way the human head.



Furthermore, for this kind of approaches we do not need to know the model of the pinna, head and others. After the acquisition, signals can be stored or directly sent to the listener. Head tracker is used to control interpolation: microphone signals are interpolated to match the orientation of listener's ears.

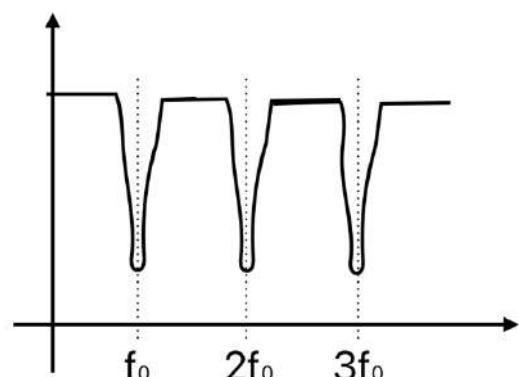
As it happens for time signals, when we sample, we bump into aliasing problems: as we know this kind of problems arise also in spatial domain. Therefore, for exact reconstruction microphones must be half a wavelength apart. Interference notches arise when this condition is not met. Interpolation produces notches (that produces some unwanted colorations) at odd multiples of

$$f_0 = \frac{Nc}{4\pi a}$$

Where:

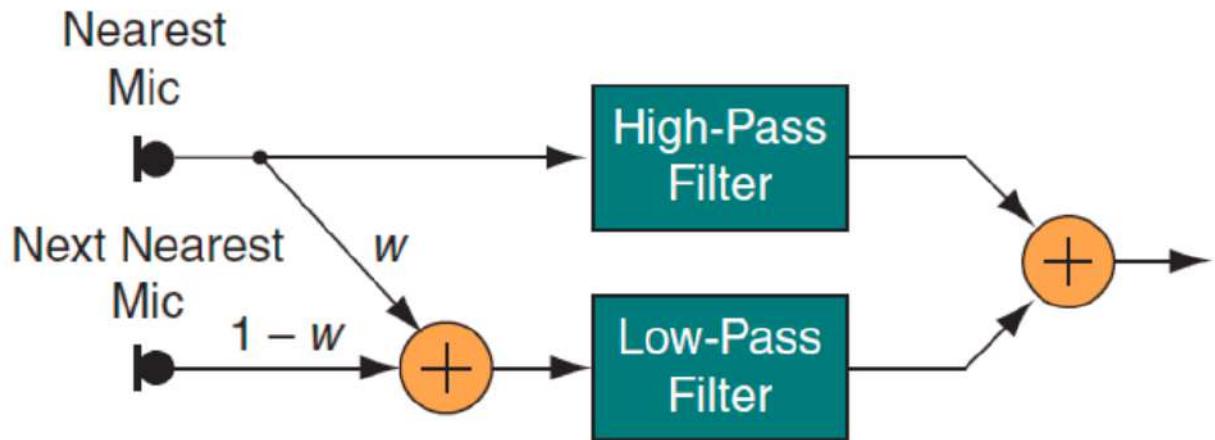
- $a$  is the radius of the microphone array
- $N$  is the number of microphones
- $c$  is the speed of sound

These notches will be present for the odds frequencies.



This means that for  $\sim 128$  microphones would be required to have  $f_0 > 20 \text{ kHz}$ . For these applications, exact reconstruction is not necessary. Usually,  $N = 8$  is acceptable for speech,  $N = 16$  is sufficient for music. Spectral notches are avoided by filtering microphone signals.

Let us see how to implement the system



The frequency that split the spectrum in low frequencies components and high frequency components is  $0.5f_0$ . Therefore:

For LF component (below  $0.5f_0$ ) interpolation is done using the two nearest microphone signals:

- $w = 0.5$  when listener's ear is halfway two microphones
- $w = 1$  when listener's ears is coincident with one of the microphones

For HF component (above  $0.5f_0$ ) the sound is directly taken from nearest microphone.

To summarise everything, we can say that

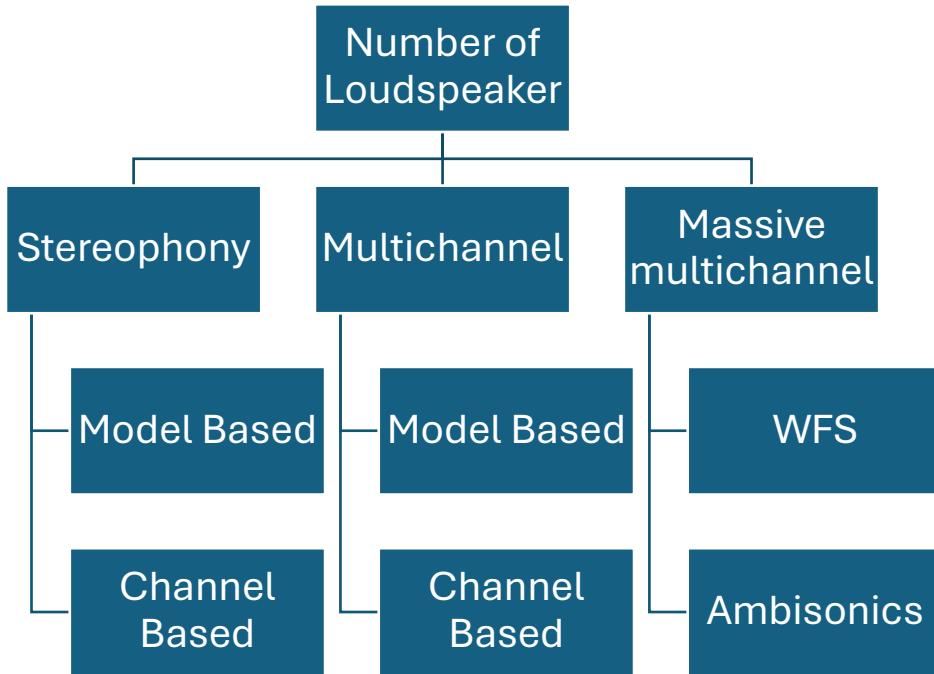
- This kind of approach is computationally simple
- It is effective for live sound (you do not have to control the number of sources and their position), faithful capturing of the acoustics of recording space.
- It does not allow the listener to move in the recording space (only head rotation)
- It does not support conventional recording practices

## References

- [1] V. R. Algazi and R. O. Duda. Headphone-based spatial sound. IEEE Signal Processing Magazine, 28(1):33–42, Jan. 2011.
- [2] P. Mackensen, U. Felderhoff, G. Theile, U. Horbach, and R. Pellegrini. Binaural room scanning - a new tool for acoustic and psychoacoustic research. J. Acoust. Soc. Am., 105(2):1343–1344, Feb. 1999.

## Spatial Sound with Loudspeakers - Stereophony and Panning

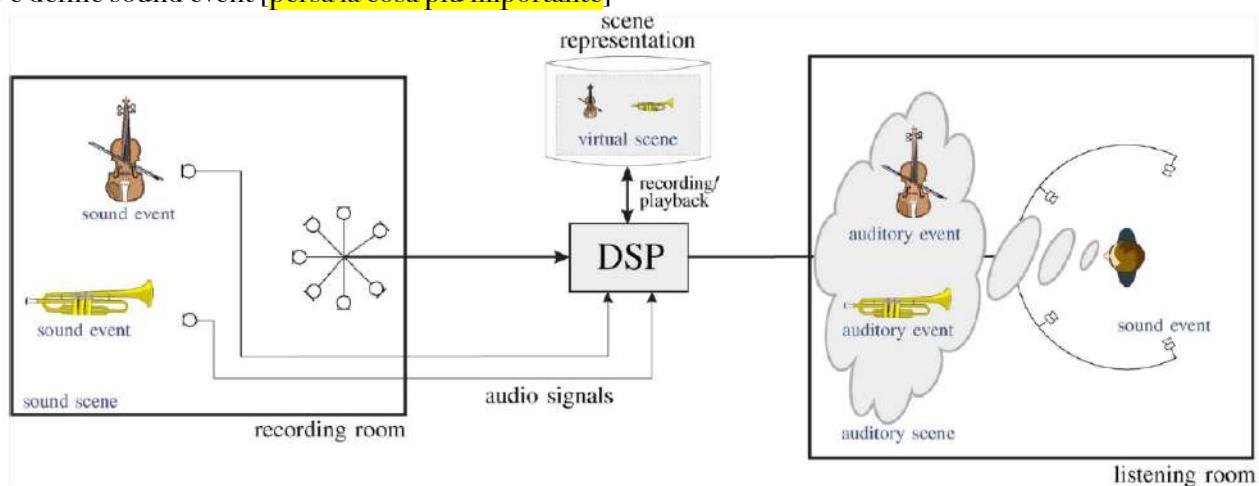
In today's lecture we will talk about stereophony and panning, basically how to create space sensation with loudspeakers. Depending on the number of loudspeakers, the technique we can used can be



First of all, we start with some definitions:

### Sound event:

We define sound event [persa la cosa più importante]



The sound event is recorder by microphone with different techniques. By means of some particular kinds of microphones we can also retrieve, from the recording, the sound scene (so how the sound events are located in space). These signals (in order to do this, we have to acquire more than one signal) are processed by DSP and reproduced in the auditory scene.

Let us now define the virtual source as an abstraction used to represent real sources (person speaking, instruments, etc.). Virtual sound scene can be composed of several virtual sources, the reproduction system

should generate a sound scene in a way that the listener will perceive the desired scene. The concept of virtual source embeds audio signal spatialization-related data (position, orientation, etc.) other parameters (level, etc.). Notice that we do not acquire just the audio signal but also some meta data.

The concept of virtual source is a very general one that can be “materialised” as:

- Plane waves
- Point sources
- Sources with a precise directivity pattern

Virtual sources can also be related to models addressing the interaction of the source with the environment:

- Wide (diffuse) sources
- Reverberation or ambience

An example could be a virtual source that represents a choir where we have many audio signals (one for each chorister) and that retrieve information also on the position of each chorister.

There are different recording styles

- **Recording style 1**  
In this kind of scenario, we record the audio signal of each source. A set of spatialisation parameters is applied in post-production. This approach is widely used in pop music recordings.
- **Recording style 2**  
With this kind of approach, a special set of microphones is used in order to record several sound events simultaneously. The spatialisation parameters and scene object are implicitly embedded into the audio tracks. This approach is commonly adopted in the production of classical music performed in good sound environments.

In practice, the two techniques are merged: signals are picked by specific arrangement of microphones in space; furthermore, spot microphones are placed close to specific instruments or instrument groups. In studio, sound editor specifies spatialisation parameters, constrained by the actual reproduction setup. In any case, independently from the technique used, the goal of sound field rendering is to translate a virtual sound scene to signals emitted by the loudspeakers in the listening room.

## Stereophony

Rendering of a sound field using two loudspeakers. There are different techniques: channel-based techniques and transform domain-based techniques.

### *Channel based*

Channel based techniques are the most widely used methods are two-channel stereo sound and multichannel (e.g. 5.1) surround sound. Obviously, they are directly related to a specific layout of loudspeakers (the reproduction set-up must be known by people that are recording). Therefore loudspeaker signals themselves must be stored and transmitted. Obviously, loudspeakers must be set up in the same layout at every reproduction stage. This system is feasible only if a limited number of loudspeakers is considered (it works just for two or not more than three loudspeakers)

### *Transform-domain based*

The signal is expressed as linear combination between bases whose weights are called coefficients of expansion

$$p(r, \omega) = \sum_{n=-\infty}^{\infty} c_n(\omega) \beta(r, \omega)$$

As we can imagine, we need a further step to decode the signal.

### *Object based technique*

(not covered in the course)

### **Two-channel stereophony**

#### *Model based stereophony*

An object that contains audio signals and meta-data used to later rendering (for example in the context of virtual reality). This is the most widely used technique. It exploits level and/or time differences between signals of loudspeakers.

$$d_1(t) = g_1 s(t - \tau_1) \quad d_2(t) = g_2 s(t - \tau_2)$$

Where:

- $s(t)$  is the signal of the virtual source
- $g_1$  and  $g_2$  are the gains
- $\tau_1$  and  $\tau_2$  are the time delays

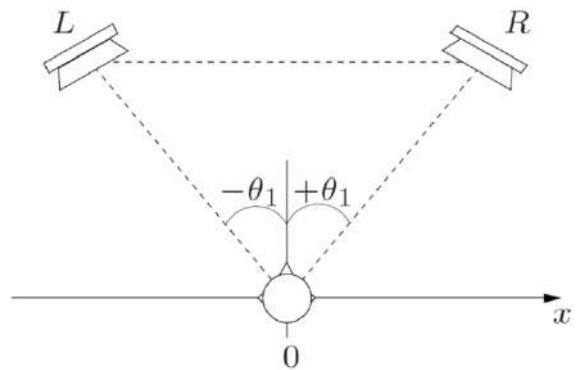
Suitable time and level differences cause the perception of virtual source in between the two loudspeakers. Amplitude panning (i.e.  $\tau_1 = \tau_2 = 0$ ) is exploited in both channel-based and model-based systems. On the other hand, time-delay panning is exploited only in channel-based systems.

Let us consider the following scenario in which the position of the loudspeaker is parametrised just by the angle theta. We assume that loudspeakers are in the far field (the front waves can be considered as plane):  
For the right loudspeaker

$$\begin{aligned} p_R(\underline{r}, \omega) &= g_R e^{j \langle \underline{k}_R, \underline{r} \rangle} \\ p_L(\underline{r}, \omega) &= g_L e^{j \langle \underline{k}_L, \underline{r} \rangle} \end{aligned}$$

Where:

- $\underline{k}_R = \frac{\omega}{c} \begin{pmatrix} \sin(\theta_1) \\ \cos(\theta_1) \end{pmatrix}$
- $\underline{k}_L = \frac{\omega}{c} \begin{pmatrix} -\sin(\theta_1) \\ \cos(\theta_1) \end{pmatrix}$



The vector  $\underline{r}$  indicates the position in space in which the sound field is evaluated

$$\underline{r} = r[x, y] \underset{\substack{\equiv \\ \text{In our specific} \\ \text{case } y=0}}{=} [x, 0]$$

Therefore, the scalar product becomes:

- $\langle \underline{k}_R, \underline{r} \rangle = \frac{\omega}{c} \sin(\theta_1) x = k_x x$
- $\langle \underline{k}_L, \underline{r} \rangle = -\frac{\omega}{c} \sin(\theta_1) x = -k_x x$

Since we are in a linear scenario and the superposition of effects holds, we can write the sound perceived as the sum of the two plane waves. By making some derivations, we obtain the

$$\begin{aligned}
 p(x, 0, \omega) &= p_R(x, 0, \omega) + p_L(x, 0, \omega) = g_R e^{jk_x x} + g_L e^{-jk_x x} \underset{\text{Euler}}{=} \\
 &= g_R (\cos(k_x x) + j \sin(k_x x)) + g_L (\cos(-k_x x) + j \sin(-k_x x)) = \\
 &= g_R \cos(k_x x) + j g_R \sin(k_x x) + g_L \cos(k_x x) - j g_L \sin(k_x x) = \\
 &= (g_R + g_L) \cos(k_x x) + j(g_R - g_L) \sin(k_x x)
 \end{aligned}$$

As we have to determinate the panning, that depends on the phase, let us evaluate the phase of the pression

$$\angle p(x, 0, \omega) = \arctan\left(\frac{(g_R - g_L) \sin(k_x x)}{(g_R + g_L) \cos(k_x x)}\right) = \arctan\left(\frac{g_R - g_L}{g_R + g_L} \tan(k_x x)\right)$$

We can recall from the Taylor expansion of the tangent that  $\tan(x) = x + \frac{x^3}{3} + o(x^5)$  that is true when *argument*  $\rightarrow 0$  (physically speaking, when the listener is very close to the origin. Ouch! The approximation of plane wave holds if we are far from the source but this one hold only if we are close to it. We need to find a trade-off for which both of them hold).

The same can be said for the  $\arctan(x)$  for which we have

$$\arctan(x) = x - \frac{x^3}{3} + o(x^5)$$

By considering this simplification, we get that

$$\angle p(x, 0, \omega) \underset{\text{Taylor}}{\approx} \frac{g_R - g_L}{g_R + g_L} k_x x = \frac{g_R - g_L}{g_R + g_L} \frac{\omega}{c} \sin(\theta_1) x$$

We remind that our goal is to approximate with a stereo system the sound field of a plane wave coming from direction  $\theta$  that we can write as

$$p_{target}(x, 0, \omega) = e^{j \frac{\omega}{c} \sin(\theta) x}$$

Whose phase is

$$\angle p_{target}(x, 0, \omega) = \frac{\omega}{c} \sin(\theta) x$$

So, in order to approximate the sensation of spatiality given by the sound wave  $p_{target}(x, 0, \omega)$  with a stereo system that creates a pressure-wave described by  $p(x, 0, \omega)$  we have to make the phases of the two signals match.

$$\frac{g_R - g_L}{g_R + g_L} \frac{\omega}{c} \sin(\theta_1) x \approx \angle p(x, 0, \omega) = \angle p_{target}(x, 0, \omega) = \frac{\omega}{c} \sin(\theta) x$$

And what we get is the **sine law of stereophony**.

$$\frac{g_R - g_L}{g_R + g_L} \sin(\theta_1) = \sin(\theta)$$

By finding the right coefficients  $g_R$  and  $g_L$ , a pair of loudspeakers at angles  $\pm \theta_1$  can create in the vicinity of the origin a local approximation of a plane wave sound field with direction of arrival  $\theta$ . Let us solve the equation in function of the ratio of gains (pan), what we can practically control.

$$\sin(\theta) = \frac{\frac{g_R - g_L}{g_R + g_L} \sin(\theta_1)}{\frac{g_R}{g_R + g_L}} = \frac{1 - \frac{g_L}{g_R}}{1 + \frac{g_L}{g_R}} \sin(\theta_1) \rightarrow \sin(\theta) \left(1 + \frac{g_L}{g_R}\right) = \left(1 - \frac{g_L}{g_R}\right) \sin(\theta_1)$$

$$\sin(\theta) + \frac{g_L}{g_R} \sin(\theta) = \sin(\theta_1) - \frac{g_L}{g_R} \sin(\theta_1) \rightarrow \frac{g_L}{g_R} \sin(\theta) + \frac{g_L}{g_R} \sin(\theta_1) = \sin(\theta_1) - \sin(\theta)$$

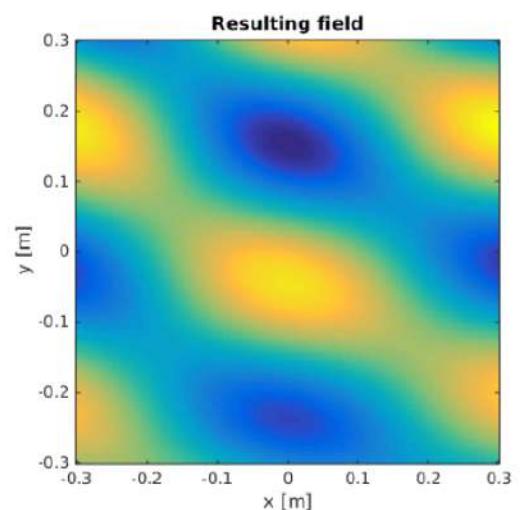
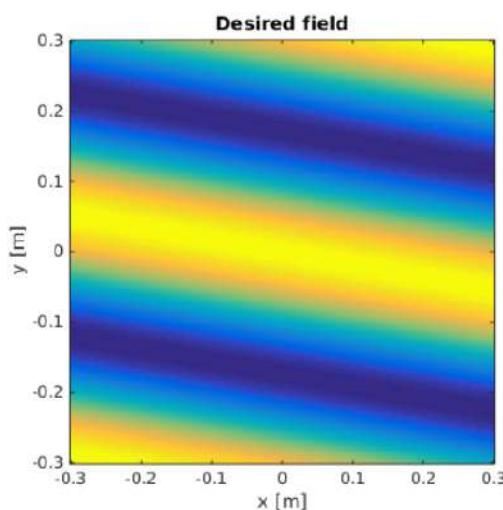
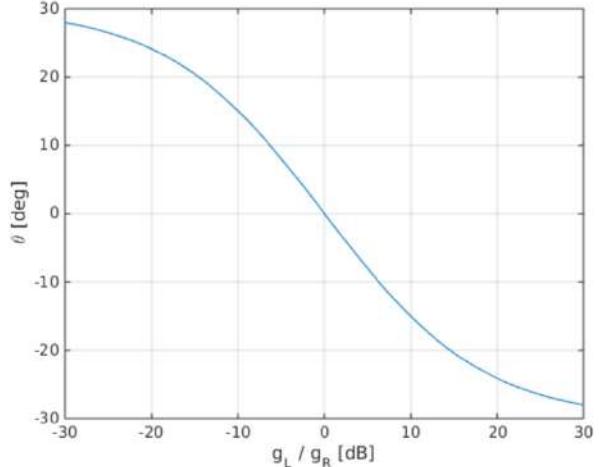
$$\frac{g_L}{g_R} = \frac{\sin(\theta_1) - \sin(\theta)}{\sin(\theta_1) + \sin(\theta)}$$

We notice that this is a ratio, so there are several combinations of  $\theta$  and  $\theta_1$  that can generate it.

Despite that the sine panning law is not linear, it can be well-approximated as a linear function around the origin. But, as we move away from the origin, we can perceive that there is an approximation.

What we have is a model representing the system parametrised by the ratio  $\frac{g_L}{g_R}$ . That is why is called model-based stereophony.

Let us now see a simulation



On the left we see the sound field generated by a source at a certain instant of time on the plane described by the two spatial variables  $x$  and  $y$ . On the right, the sound field reconstructed by two loudspeakers. As we can notice, stereophony allows a coarse approximation in a very small region around the origin (sweet spot).

### Channel based stereophony

Pay attention that there is a difference between channel stereophony and channel based technique for channel stereophony: in the former we are talking about the rendering of the reproduction environment while in the latter about recording.

The idea is: we want to create level differences using microphone directivity and create time differences exploiting microphone distance. The beam pattern can be represented with the following expression:

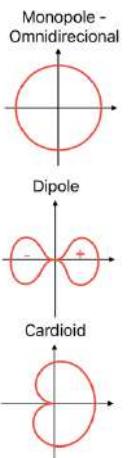
$$BP_N(\theta) = a_0 + \sum_{n=1}^N a_n \cdot \cos^n(\theta)$$

Where  $N$  is the order of our beam pattern. But what is a beam pattern? The beam pattern is a periodic function of  $\theta$  used to identify the source direction. We can normalise the function by writing

$$a_0 = 1 - \sum_{n=1}^N a_n$$

For a first order beam pattern we have

$$BP_1 = (1 - a_1) + a_1 \cdot \cos(\theta) = \begin{cases} a_1 = 1 \rightarrow BP = \cos(\theta) \\ a_1 = 0.5 \rightarrow BP = \frac{1}{2} + \frac{1}{2}\cos(\theta) \end{cases}$$



Therefore, the beam pattern for a difference of two angles in the case of  $a_1 = 1$  is:

$$BP_1(\theta - \theta_s) = \cos(\theta - \theta_s) = \cos(\theta) \cos(\theta_s) - \sin(\theta) \sin(\theta_s)$$

As we have seen before, the  $\cos(\theta)$  can be seen as a dipole along the x axis, while the  $\sin(\theta)$  can be seen as a dipole along the y axis. The former is weighted by  $\cos(\theta_s)$  and the latter by  $\sin(\theta_s)$ .

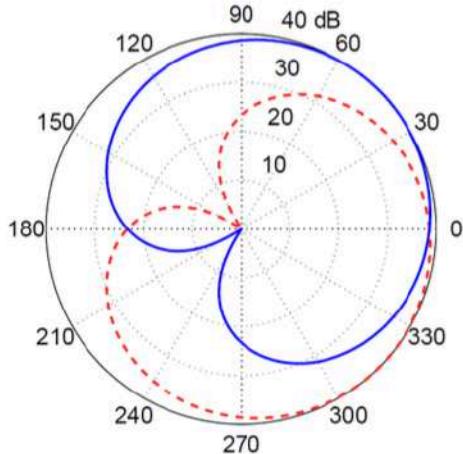
If we write the complete formula, we get

$$BP_1(\theta - \theta_s) = (1 - a) + a[\cos(\theta) \cos(\theta_s) - \sin(\theta) \sin(\theta_s)]$$

That is a quite intuitive result! The beam pattern can be seen as a combination of two properly weighted base functions! Let us see how this theory is practically implemented.

### *XY Microphone configuration*

Two microphones with cardioid pattern.



(a) Patterns.

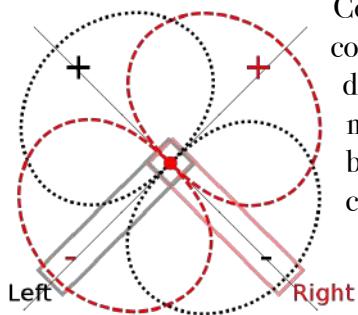


(b) Photo.

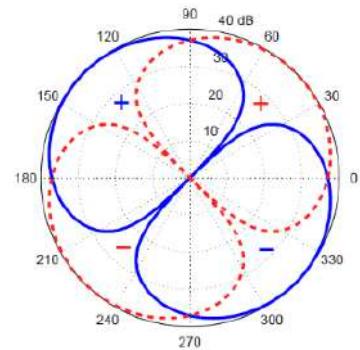
The level difference is obtained by a proper orientation of directional microphones. As they are placed in the same position, there is no time difference to record.

### *Blumlein Pair*

The Blumlein Pair has been invented by Alan Blumlein. It is very interesting reading about his life, and it is quite curious, especially looking at the pattern, noticing that that in German Blümlein means "little flower".



Coming back to engineering, the configuration is the same as before but with different type of microphones: gradient microphones whose pattern is represented by a kind of dipole shape. Their directivity can be mathematically described by



$$F(\alpha) = \cos\left(\alpha \pm \frac{\pi}{4}\right)$$

In this kind of scenario, a source placed at  $\alpha$  (with respect to stereo pair axis) emitting signal  $S(\omega)$  is perceived by the two microphones as

$$x_L(\omega) = g_L S(\omega) = \cos\left(\alpha + \frac{\pi}{4}\right) S(\omega) \rightarrow g_L = \cos\left(\alpha + \frac{\pi}{4}\right)$$

$$x_R(\omega) = g_R S(\omega) = \cos\left(\alpha - \frac{\pi}{4}\right) S(\omega) \rightarrow g_R = \cos\left(\alpha - \frac{\pi}{4}\right)$$

If we substitute them in the law of stereophony, we get that

$$\sin(\theta) = \frac{g_R - g_L}{g_R + g_L} \sin(\theta_1)$$

$$\sin(\theta) = \frac{\cos\left(\alpha - \frac{\pi}{4}\right) - \cos\left(\alpha + \frac{\pi}{4}\right)}{\cos\left(\alpha - \frac{\pi}{4}\right) + \cos\left(\alpha + \frac{\pi}{4}\right)} \sin(\theta_1) \stackrel{\substack{\text{Euler because with goniometry} \\ \text{mammina mammina}}}{=} \frac{\cos(\alpha) = \frac{e^{j\alpha} + e^{-j\alpha}}{2}}{2}$$

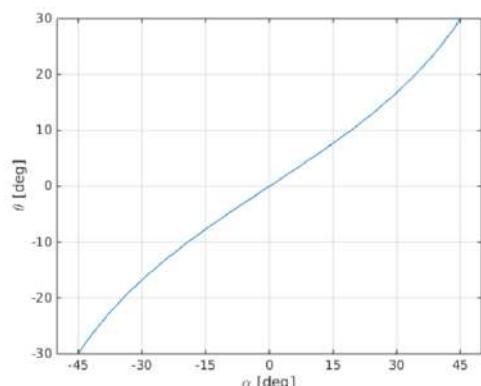
$$\begin{aligned}
&= \frac{e^{j(\alpha-\frac{\pi}{4})} + e^{-j(\alpha-\frac{\pi}{4})} - e^{j(\alpha+\frac{\pi}{4})} - e^{-j(\alpha+\frac{\pi}{4})}}{2} \sin(\theta_1) = \\
&= \frac{e^{j\alpha} e^{-j\frac{\pi}{4}} + e^{-j\alpha} e^{j\frac{\pi}{4}} - e^{j\alpha} e^{j\frac{\pi}{4}} - e^{-j\alpha} e^{-j\frac{\pi}{4}}}{2} \sin(\theta_1) = \\
&= \frac{-e^{j\alpha} \left( e^{j\frac{\pi}{4}} - e^{-j\frac{\pi}{4}} \right) + e^{-j\alpha} \left( e^{j\frac{\pi}{4}} - e^{-j\frac{\pi}{4}} \right)}{e^{j\alpha} \left( e^{j\frac{\pi}{4}} + e^{-j\frac{\pi}{4}} \right) + e^{-j\alpha} \left( e^{j\frac{\pi}{4}} + e^{-j\frac{\pi}{4}} \right)} \sin(\theta_1) = \\
&= -\frac{e^{j\alpha} \left( e^{j\frac{\pi}{4}} - e^{-j\frac{\pi}{4}} \right) - e^{-j\alpha} \left( e^{j\frac{\pi}{4}} - e^{-j\frac{\pi}{4}} \right)}{e^{j\alpha} \left( e^{j\frac{\pi}{4}} + e^{-j\frac{\pi}{4}} \right) + e^{-j\alpha} \left( e^{j\frac{\pi}{4}} + e^{-j\frac{\pi}{4}} \right)} \sin(\theta_1) = \\
&= -\frac{(2j)^2 \frac{e^{j\frac{\pi}{4}} - e^{-j\frac{\pi}{4}}}{2j} \frac{e^{j\alpha} - e^{-j\alpha}}{2j}}{\left( e^{j\frac{\pi}{4}} + e^{-j\frac{\pi}{4}} \right) \left( e^{j\alpha} + e^{-j\alpha} \right)} \sin(\theta_1) = \\
&= -\frac{(2j)^2 \frac{e^{j\frac{\pi}{4}} - e^{-j\frac{\pi}{4}}}{2j} \frac{e^{j\alpha} - e^{-j\alpha}}{2j}}{2^2 \frac{e^{j\frac{\pi}{4}} + e^{-j\frac{\pi}{4}}}{2} \frac{e^{j\alpha} + e^{-j\alpha}}{2}} \sin(\theta_1) = \frac{\sin\left(\frac{\pi}{4}\right) \sin(\alpha)}{\cos\left(\frac{\pi}{4}\right) \cos(\alpha)} \sin(\theta_1) = \frac{\frac{\sqrt{2}}{2} \sin(\alpha)}{\frac{\sqrt{2}}{2} \cos(\alpha)} \sin(\theta_1) = \\
&= \tan(\alpha) \sin(\theta_1)
\end{aligned}$$

Therefore, the angle of the sound coming from a real source can be approximated as the angle of the DOA of the reproduction sound multiplied by a certain weight equal to  $\tan(\alpha)$ .

$$\sin(\theta) = \tan(\alpha) \sin(\theta_1)$$

Where:

- $\alpha$  is the angle of the source relative to the microphone pair
- $\theta_1$  is the angle of the loudspeaker relative to the listener
- $\theta$  is the angle of the generated virtual source relative to the listener

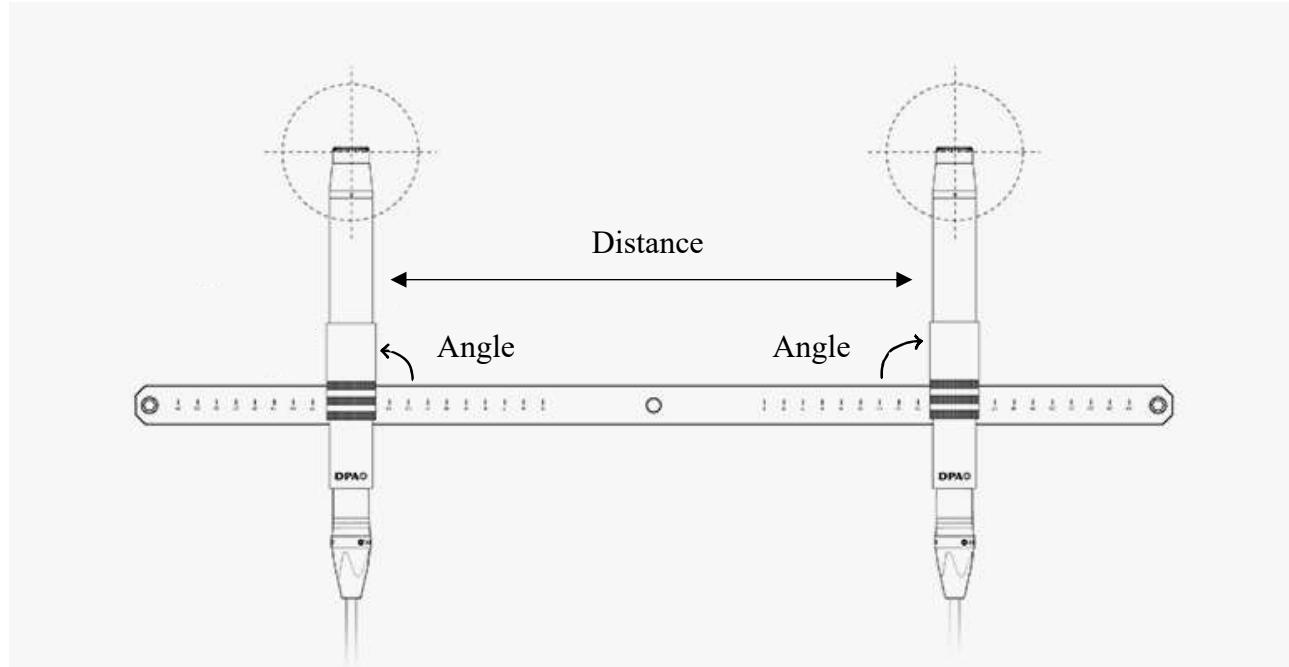


### *Other XY configurations*

The analysis of other XY systems (with cardioid, super cardioid, hyper cardioid microphones) is similar but more difficult. In any case, stereophony is effective only in a very small region around the ideal listener position as the plane wave approximation does not hold anymore if we move around the sweet spot position.

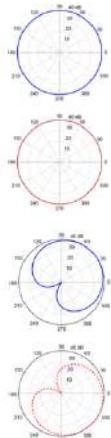
### *AB Microphone configuration*

Let us see two different AB microphone configurations. Firstly, in the image below, the representation.



### *AB pair with omnidirectional microphones*

In this type of approach, the time difference is generated by the distance of the microphones. The level difference is very hard to detect as we work with omnidirectional microphones that are very close to each other. Furthermore, we can bump into the so-called comb filter effect, an interference at high frequency.



### *AB pair with cardioid microphones*

In this type of approach, the time difference is again generated by the distance of the microphones. With cardioid microphones, that have a specific directivity, the level difference is more easily detectable.

This last implementation has been widely personalised by different companies in the past. We report a list of the most famous ones.

Method	Distance	Angle	Notes
NOS	30 cm	90°	
OLSON	20 cm	135°	Designed by Leon Nelson in Aduionics. It suffers of indistinct sound in front and in the middle due to the big angle. It is good if sources are placed at the side
ORTF	17 cm	110°	The one of the French Broadcasting Office, great to transform a signal from mono to stereo.
RAI	21 cm	100°	The one of mamma RAI.
DIN	20 cm	90°	Used for piano and violin recordings.

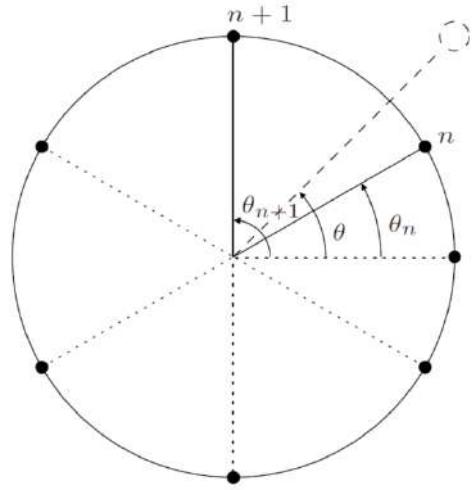
## Multichannel stereophony

### *Model based stereophony*

We will consider now a large number of loudspeakers. In this context, in fact, extended panning range is obtained by adding more loudspeakers. Since the scenario is changed, we have to derive a new panning law valid for a larger number of loudspeakers. We come up with the vector-based amplitude panning (VBAP):

for each virtual source we determine the active pair of loudspeakers and their panning function. Simply speaking, if we want to reproduce the sound of a person that is walking around us, the loudspeakers will turn on and off circularly in order to recreate the sensation. This kind of approach can be extended also to 3D scenarios

The scenario is the one reported on the right: the listener is placed in the origin of the circumference; the black dots are the loudspeakers characterised by an index  $n$  and an angle  $\theta_n$ .



As before, we have to assume that the loudspeakers are far from the listener in order to use the plane wave assumption. What we want to do is to generate a virtual source

$$\underline{v} = [\cos(\theta), \sin(\theta)]^T$$

It can be generated by multiplying a vector  $\underline{g}$  containing the two gains

$$\underline{g} = [g_n, g_{n+1}]^T$$

And then the coordinates of the two active loudspeakers

$$\underline{\underline{M}} = \begin{pmatrix} \cos(\theta_n) & \cos(\theta_{n+1}) \\ \sin(\theta_n) & \sin(\theta_{n+1}) \end{pmatrix}$$

Hence, the position of the virtual source can be related to the position of the loudspeaker by the gains

$$\underline{v} = \underline{\underline{M}} \underline{g}$$

Gains that can be found by inverting the relation

$$\begin{pmatrix} g_n \\ g_{n+1} \end{pmatrix} = \underline{\underline{M}}^{-1} \underline{v}$$

Where:

$$\underline{\underline{M}}^{-1} = \frac{1}{\det(M)} \begin{pmatrix} \sin(\theta_{n+1}) & -\cos(\theta_{n+1}) \\ -\sin(\theta_n) & \cos(\theta_n) \end{pmatrix}$$

Where the determinant is expressed by

$$\begin{aligned}
 \det(M) &= \cos(\theta_n) \sin(\theta_{n+1}) - \sin(\theta_n) \cos(\theta_{n+1}) = \\
 &= \frac{1}{2} [\sin(\theta_{n+1} + \theta_n) + \sin(\theta_{n+1} - \theta_n)] - \frac{1}{2} [\sin(\theta_n + \theta_{n+1}) + \sin(\theta_n - \theta_{n+1})] = \\
 &= \frac{1}{2} \sin(\theta_{n+1} + \theta_n) + \frac{1}{2} \sin(\theta_{n+1} - \theta_n) - \frac{1}{2} \sin(\theta_n + \theta_{n+1}) - \frac{1}{2} \sin(\theta_n - \theta_{n+1}) = \\
 &= \frac{1}{2} \sin(\theta_{n+1} + \theta_n) + \frac{1}{2} \sin(\theta_{n+1} - \theta_n) - \frac{1}{2} \sin(\theta_{n+1} + \theta_n) + \frac{1}{2} \sin(\theta_{n+1} - \theta_n) = \\
 &= \sin(\theta_{n+1} - \theta_n) = \Delta
 \end{aligned}$$

At the end, what we get is

$$\begin{aligned}
 \begin{pmatrix} g_n \\ g_{n+1} \end{pmatrix} &= \begin{pmatrix} \frac{\sin(\theta_{n+1})}{\Delta} & -\frac{\cos(\theta_{n+1})}{\Delta} \\ -\frac{\sin(\theta_n)}{\Delta} & \frac{\cos(\theta_n)}{\Delta} \end{pmatrix} \underline{v} = \begin{pmatrix} \frac{\sin(\theta_{n+1}) \cos(\theta) - \cos(\theta_{n+1}) \sin(\theta)}{\Delta} \\ \frac{-\sin(\theta_n) \cos(\theta) + \cos(\theta_n) \sin(\theta)}{\Delta} \end{pmatrix} = \\
 &= \begin{pmatrix} \frac{\sin(\theta_{n+1}) \cos(\theta) - \cos(\theta_{n+1}) \sin(\theta)}{\Delta} \\ \frac{-\sin(\theta_n) \cos(\theta) + \cos(\theta_n) \sin(\theta)}{\Delta} \end{pmatrix} = \\
 &= \begin{pmatrix} \frac{\frac{1}{2} [\sin(\theta_{n+1} + \theta) + \sin(\theta_{n+1} - \theta)] - \frac{1}{2} [\sin(\theta + \theta_{n+1}) + \sin(\theta - \theta_{n+1})]}{\Delta} \\ \frac{-\frac{1}{2} [\sin(\theta_n + \theta) + \sin(\theta_n - \theta)] + \frac{1}{2} [\sin(\theta + \theta_n) + \sin(\theta - \theta_n)]}{\Delta} \end{pmatrix} = \\
 &= \begin{pmatrix} \frac{\frac{1}{2} \sin(\theta_{n+1} + \theta) + \frac{1}{2} \sin(\theta_{n+1} - \theta) - \frac{1}{2} \sin(\theta + \theta_{n+1}) - \frac{1}{2} \sin(\theta - \theta_{n+1})}{\Delta} \\ \frac{-\frac{1}{2} \sin(\theta_n + \theta) - \frac{1}{2} \sin(\theta_n - \theta) + \frac{1}{2} \sin(\theta + \theta_n) + \frac{1}{2} \sin(\theta - \theta_n)}{\Delta} \end{pmatrix} = \\
 &= \begin{pmatrix} \frac{\sin(\theta_{n+1} - \theta)}{\Delta} \\ \frac{\sin(\theta - \theta_n)}{\Delta} \end{pmatrix} = \begin{pmatrix} \frac{\sin(\theta_{n+1} - \theta)}{\sin(\theta_{n+1} - \theta_n)} \\ \frac{\sin(\theta - \theta_n)}{\sin(\theta_{n+1} - \theta_n)} \end{pmatrix}
 \end{aligned}$$

Therefore, the coefficients will be

$$g_n(\theta) = \frac{\sin(\theta_{n+1} - \theta)}{\sin(\theta_{n+1} - \theta_n)}$$

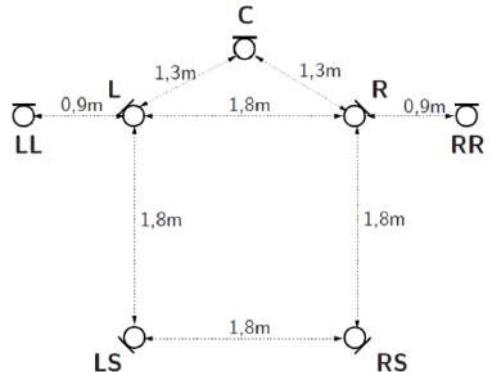
$$g_{n+1}(\theta) = \frac{\sin(\theta - \theta_n)}{\sin(\theta_{n+1} - \theta)}$$

This is the gain that we have to use to represent a source in the position  $\theta$  that is  $\theta_n \leq \theta \leq \theta_{n+1}$ . Obviously, the others are set to zero. By iterating the reasoning, we get that, in general,

$$g_v(\theta) = \begin{cases} \frac{\sin(\theta_{n+1} - \theta)}{\sin(\theta_{n+1} - \theta_n)}, & \text{for } v = n \\ \frac{\sin(\theta - \theta_n)}{\sin(\theta_{n+1} - \theta)}, & \text{for } v = n + 1 \\ 0, & \text{otherwise} \end{cases}$$

### *Channel based stereophony*

The approach is very simple: we recreate the exact arrangement of the microphones with the loudspeakers. An example is the 5.1 surround implemented with the technique called Fukada tree by means of cardioid microphones. It is represented on the right.



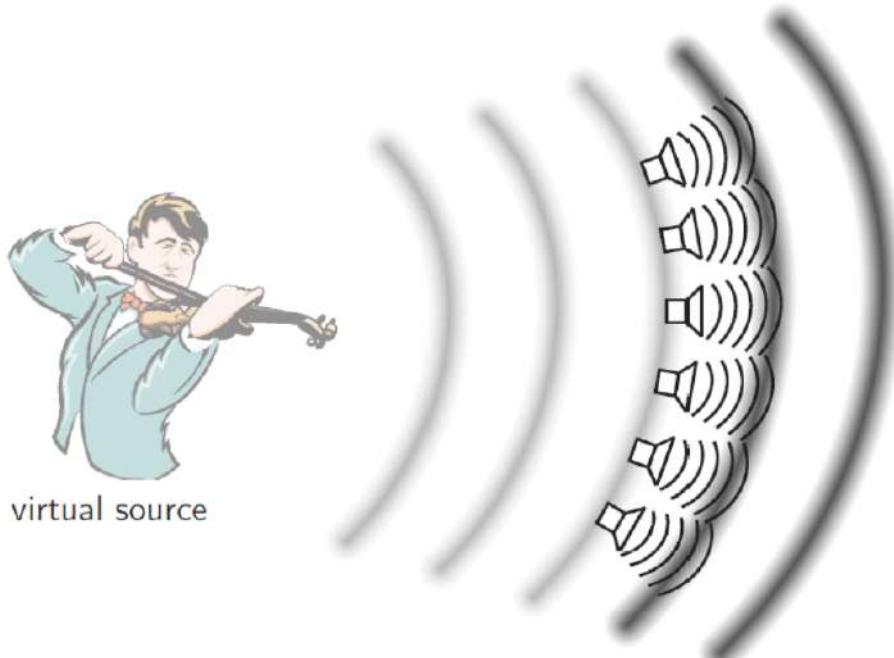
### References

- [1] S. Spors, H. Wierstorf, A. Raake, F. Melchior, M. Frank, and F. Zotter. Spatial sound with loudspeakers and its perception: A review of the current state. *Proceedings of the IEEE*, 101(9):1920–1938, 2013

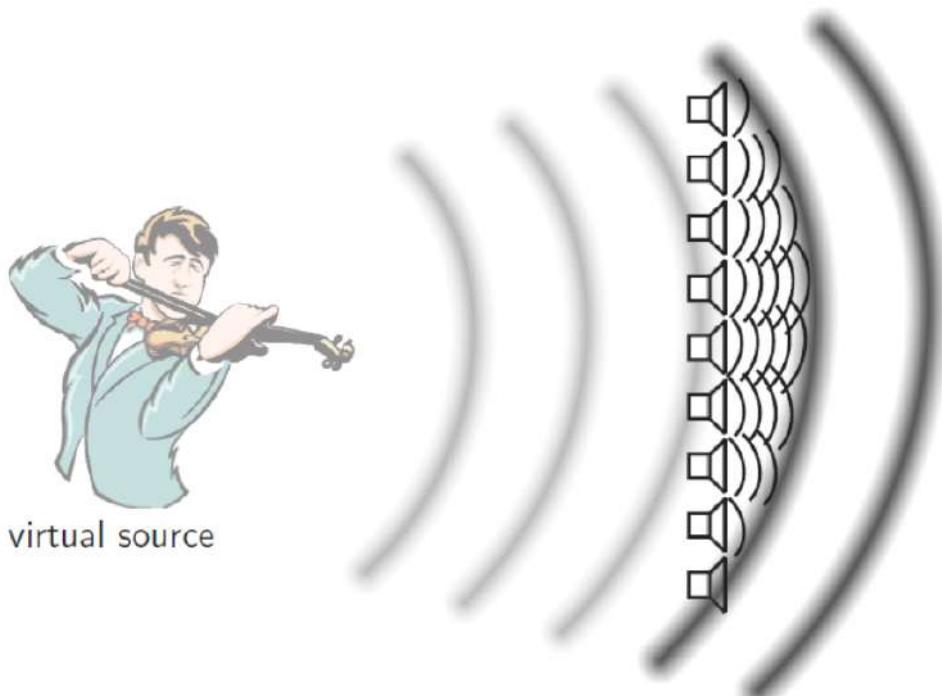
### Huygens' principle

Considering a source that emits sinewaves, we can model the front wave as a continuous distribution of sources (called secondary sources) that emits spherical waveforms.

The idea is: by properly driving these secondary sources, it is possible to obtain the original sound field produced by the original source (called virtual source). This is because the secondary sources mutual interfere generating the sound field that resembles the one of the virtual source.



It is also possible to push further this concept by distributing the secondary sources in a shape that is not exactly the one of the front wave that we are considering by modulating the amplitude.

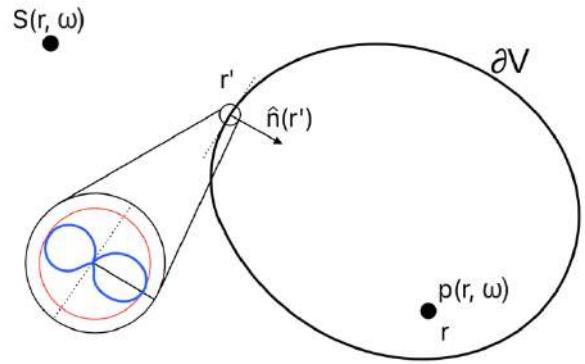


## Kirchhoff Helmholtz integral

Let us review the Kirchhoff Helmholtz integral as a generalisation of the Huygens' principle. We consider a 3D volume in the environment characterised by a boundary called  $\partial V$ . We consider an arbitrary point inside the volume and we call it  $A$ . Then an external source  $s$  generating a sound at frequency  $\omega$  in the position  $r$ .

The Kirchhoff Helmholtz integral tells us that we can represent the sound wave by exploiting a continuous distribution of secondary sources placed on the boundary of the volume that we are considering. We call a generic point on the boundary  $r'$ . We are interesting to find the normal component to this point called  $\hat{n}(r')$  (versor orthogonal to the tangent in that point). Let us zoom on our boundary. We consider that in that point there are two sources: a monopole component (in red) and a dipole component (in blue) directed along the versor  $\hat{n}(r')$ . By properly waiting the monopole component and the dipole one for every point, we can represent the sound field inside the volume we are considering.

Mathematically speaking,



$$-\oint_{\partial V} \left[ \frac{\partial}{\partial \hat{n}} p(\underline{r}', \omega) G(\underline{r} | \underline{r}', \omega) - p(\underline{r}', \omega) \frac{\partial}{\partial \hat{n}} G(\underline{r} | \underline{r}', \omega) \right] dA(\underline{r}') = \begin{cases} p(\underline{r}, \omega), & \underline{r} \in V \\ 0, & \underline{r} \notin V \end{cases}$$

In this expression, the derivation with respect to  $\hat{n}$  makes the monopole directional becoming a dipole.

We remind that this result lies on the following hypothesis:

- No sources inside the volume
- The distribution of sources is placed on the boundary
- The virtual source it is outside the volume

Obviously, a model like this one is impossible to realise practically. In order to realise it we have to find some solutions

Problem	Solution	Notes
Two kinds of sources at the boundary: monopoles and dipoles	Elimination of the dipoles	It is quite impossible to realise a perfect dipole with a loudspeaker. Furthermore, it is practically impossible to have two different speakers in the same position.
Generation of sound signals on the whole surface	Distribution of point sources only on a closed curve on the horizontal plane	Can you imagine a sphere made of loudspeakers?
Generation of a continuous source distribution	Discrete sources (loudspeakers)	We perform a sort of spatial sampling of the continuous distribution we are considering.
Determination of the loudspeaker signals	Evaluation of acoustic source models	How do we drive the loudspeakers? We assume that a loudspeaker responds to an ideal and specific acoustic model (spherical or planar waves)

Let us apply the solutions of the table into the Green's function. Firstly, we eliminate the dipoles by imposing

$$\frac{\partial}{\partial \hat{n}} G(\underline{r} | \underline{r}' \cdot \omega) \Big|_{\underline{r}' \in \partial \mathcal{V}} = 0$$

So, this is the homogeneous Neumann condition imposed on  $\partial \mathcal{V}$  where the boundary  $\partial \mathcal{V}$  is modified as a rigid surface. We assume that the gradient of the pressure field along the direction of the verson  $n$  is equal to zero.

$$\langle \nabla p(\underline{r}', \omega), \hat{n}(\underline{r}') \rangle = 0$$

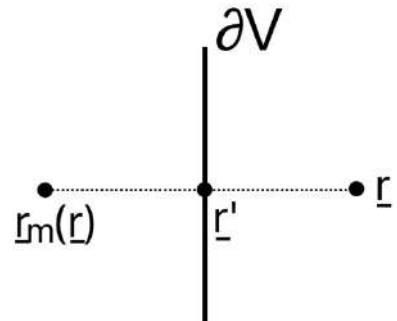
We can express the desired Neumann Green's function deriving it from a suitable homogeneous solution to the free field Green's function  $G_0$  that depends on the geometry of  $\partial \mathcal{V}$  and it is expressed for linear/planar geometry

$$G_N(\underline{r} | \underline{r}', \omega) = 2G_0(\underline{r} | \underline{r}', \omega)$$

Why so? This is the response of a source placed in the position  $\underline{r}'$  and evaluated in the position  $\underline{r}$  due to the Newman condition. How can we find a solution?

By considering a mirrored point symmetric to  $\underline{r}$  with respect to  $\underline{r}'$ . Since it is symmetric, we can write it as it is written above. In fact, In the assumption of the free field homogeneous Newman boundary condition, this is equal to

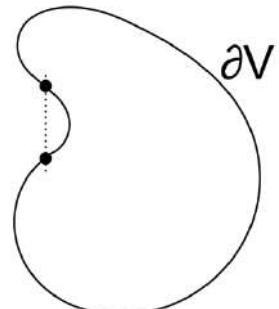
$$\begin{aligned} G_N(\underline{r} | \underline{r}', \omega) &= G_0(\underline{r} | \underline{r}', \omega) + G_0(\underline{r}_m(\underline{r}) | \underline{r}', \omega) \\ &\quad \stackrel{|\underline{r}' - \underline{r}| = |\underline{r}_m(\underline{r}) - \underline{r}'|}{=} \\ &= G_0(\underline{r} | \underline{r}', \omega) + G_0(\underline{r} | \underline{r}', \omega) = 2G_0(\underline{r} | \underline{r}', \omega) \end{aligned}$$



This is the solution that satisfies the free field homogeneous Newman boundary. We underline that we considered a linear geometry to reach this result (the boundary on which the point is mirrored is a line).

Of course, this is an assumption, and it will introduce some drawbacks. Anyway, we do another approximation: we consider that this result holds also for arbitrary shaped contours. Hence, we accept that sound field outside  $\mathcal{V}$  is not zero. Actually, boundary  $\partial \mathcal{V}$  must be convex to avoid contributions from outside  $\mathcal{V}$  to propagate inside  $\mathcal{V}$ . If the source is not complex, it is possible that some contributions of loudspeakers located in different points generate interferences with the sound field produced by the others as it happens in the image on the right.

Last drawback is that we have to accept is that the reproduced sound field does not match perfectly the desired one, neither in  $\mathcal{V}$  as the reproduced field is the superposition of desired field and undesired reflections (due to the rigid modelling of the boundary).



By considering this simplification, the integral reduces to this formula

$$\begin{aligned} p(\underline{r}, \omega) &= -\oint_{\partial V} \left[ \frac{\partial}{\partial \hat{n}} p(\underline{r}', \omega) G(\underline{r}|\underline{r}', \omega) - p(\underline{r}', \omega) \underbrace{\frac{\partial}{\partial \hat{n}} G(\underline{r}|\underline{r}', \omega)}_{=0} \right] dA(\underline{r}') \approx \\ &\approx -\oint_{\partial V} D(\underline{r}, \omega) G_0(\underline{r}|\underline{r}', \omega) dS_0(\underline{r}') \end{aligned}$$

Where  $D(\underline{r}, \omega)$  is the secondary source driving function

$$D(\underline{r}, \omega) = 2a(\underline{r}') \frac{\partial}{\partial \hat{n}} p(\underline{r}', \omega)$$

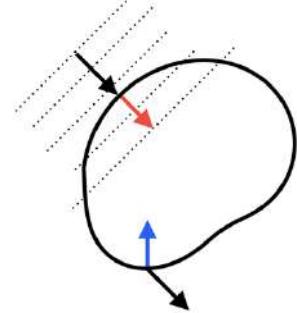
And  $a(\underline{r}')$  is a suitable selection function that depends on the position/direction of the virtual source. For example, for a field of plane wave

$$p(\underline{r}, \omega) = A(\omega) e^{j\langle \underline{k}_0, \underline{r} \rangle}$$

The selection function is expressed by

$$a(\underline{r}') = \begin{cases} 1, & \text{if } \langle \underline{k}_0, \hat{n}(\underline{r}') \rangle > 0 \\ 0, & \text{otherwise} \end{cases}$$

This means that it is “turned on” only if the inner product is greater than zero. Just to have an intuition about this: for the point in red, surely the inner product between the impinging wave and the normal is higher than 0 (switched on), for the one in blue it is surely less than zero (switched off).



Another example, for a point source, the selection function can be expressed as it follows. Considering that the field of a point source positioned at  $\underline{z}$ , the sound field can be express as

$$p(\underline{r}, \omega) = A(\omega) \frac{e^{-j\frac{\omega}{c}\|\underline{r}-\underline{z}\|}}{4\pi\|\underline{r}-\underline{z}\|}$$

The selection function will be expressed as

$$a(\underline{r}') = \begin{cases} 1, & \text{if } \langle \underline{r}' - \underline{z}, \hat{n}(\underline{r}') \rangle > 0 \\ 0, & \text{otherwise} \end{cases}$$

### Derivation of wave field synthesis – 3D Wave field Synthesis

What we have to do now, is to derive the driving function for distribution of secondary sources. This driving functions are modelled in different ways depending on the dimensionality that we want to consider. We start from the same scenario: we considered for the Helmholtz integral and a Green's function that could determinate the characteristics of the secondary sources:

$$G_{3D}(\underline{r}|\underline{r}', \omega) = \frac{e^{-j\frac{\omega}{c}\|\underline{r}-\underline{r}'\|}}{4\pi\|\underline{r}-\underline{r}'\|}$$

$G_{3D}(\underline{r}|\underline{r}', \omega)$  is the field of a point source with monopole (omnidirectional) characteristics positioned at  $\underline{r}'$ .

3D WFS can be realised by surrounding the listening volume  $\mathcal{V}$  by a continuous distribution of point sources on its boundary  $\partial\mathcal{V}$ .

3D WFS can be realised by surrounding the listening volume  $\mathcal{V}$  by a continuous distribution of point sources on its boundary  $\partial\mathcal{V}$ . Secondary sources are driven by a driving function  $D(\underline{r}', \omega)$ . Once again, its explicit form depends on virtual source model and on the geometry of the sound reproduction system.

We start from a simple case of a plane wave virtual field. The driving function can be written as

$$D_{pw,3D}(\underline{r}', \omega) = 2a(\underline{r}')j\omega A(\omega) \frac{\langle \hat{k}_0, \hat{n}(\underline{r}') \rangle}{c} e^{j\langle \hat{k}_0, \underline{r}' \rangle}$$

Inverse Fourier over time returns

$$d_{pw,3D}(\underline{r}', t) = 2a(\underline{r}') \frac{\langle \hat{k}_0, \hat{n}(\underline{r}') \rangle}{c} \frac{\partial}{\partial t} A\left(t - \frac{\langle \hat{k}_0, \underline{r}' \rangle}{c}\right)$$

The computation is efficient in time domain as the function it is nothing but the weighted derivative of the delayed source signal. The derivative can be performed by filtering the signal with a  $j\omega$ -characteristics. For what it concerns the point source, the driving function can be represented by

$$D_{ps,3D}(\underline{r}', \omega) = -2a(\underline{r}') \frac{\langle \underline{r}' - \underline{z}, \hat{n}(\underline{r}') \rangle}{\|\underline{r}' - \underline{z}\|^2} \times \left( \frac{1}{\|\underline{r}' - \underline{z}\|} + \frac{j\omega}{c} \right) A(\omega) e^{-j\frac{\omega}{c}\|\underline{r}' - \underline{z}\|}$$

That in time domain becomes

$$d_{ps,3D}(\underline{r}', t) = -2a(\underline{r}') \frac{\langle \underline{r}' - \underline{z}, \hat{n}(\underline{r}') \rangle}{\|\underline{r}' - \underline{z}\|^2} \times \left( \frac{1}{\|\underline{r}' - \underline{z}\|} + \frac{1}{c} \frac{\partial}{\partial t} \right) A\left(t - \frac{\|\underline{r}' - \underline{z}\|}{c}\right)$$

Let us focus on the planar distribution of secondary sources considering that any surface can degenerate into an infinite plane (for the same principle that if we zoom in on a curve, it seems a straight line). This plane will divide the inside of a volume from the outside of the volume. We are interested in the inside of the plane. Assuming a planar distribution of point sources on the  $xz$  plane, the reproduced sound field is

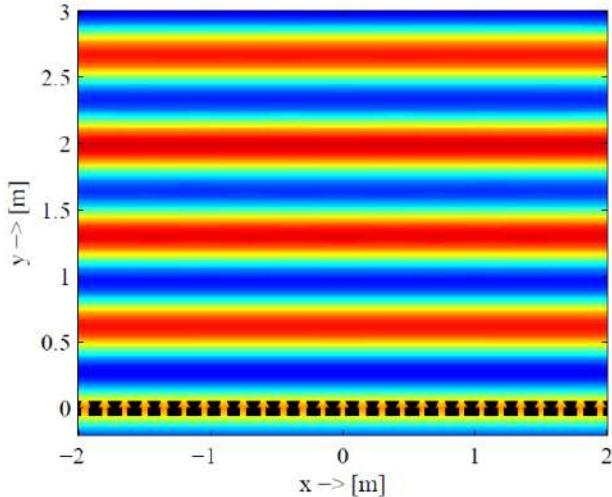
$$p(\underline{r}, \omega) = \iint_{-\infty}^{\infty} D_{3D}(\underline{r}', \omega) G_{3D}(\underline{r}|\underline{r}', \omega) dx' dz'$$

What we have is first Rayleigh integral. Reproduction is only correct in one of the two half volumes separated by the secondary source distribution.

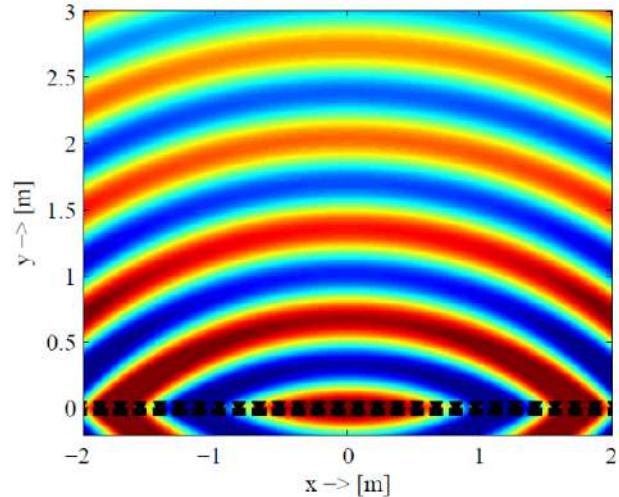
In practice we have, of course, a discrete distribution of point sources which is basically a planar distribution of loudspeakers. This makes some artefacts arise:

- Spatial aliasing
- Truncation: diffraction phenomena in the context of waves synthesis.

If we look at the system from the top (reminding that we are not dealing with an array of loudspeaker but with a surface of loudspeakers), what we get is



(a) plane wave ( $f_{pw} = 500$  Hz,  $\mathbf{n}_{pw} = [0 \ 1 \ 0]^T$ )



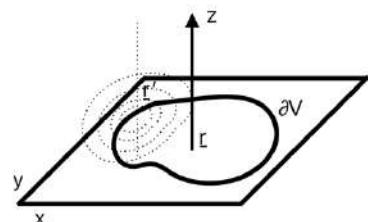
(b) spherical wave ( $f_{sw} = 500$  Hz,  $\mathbf{x}_S = [0 \ -2 \ 0]^T$  m)

In the second case, we have a point source model placed in  $y = -2$ . Inside the volume, we have a good reproduction of the soundfield but outside the volume the sound field is not zero at all. Furthermore, we have a decay in amplitude in the case of the spherical waves.

### Derivation of wave field synthesis - 2D Wave field Synthesis

When we think about a real sound environment, we imagine that the loudspeakers are placed on the same plane of the ears. Therefore, a 2D wave field can be surely more interesting. In this scenario, the source propagates “circular” plane waves along an axis. Obviously, with this kind of system we will not be able to reproduce sounds that comes from above or below the plane. The secondary sources will be disposed on a line and not more on a plane and the Green’s function that will represent the system will be

$$G_{2D}(\underline{r}|\underline{r}', \omega) = \frac{j}{4} H_0^{(2)}\left(\frac{\omega}{c} \|\underline{r} - \underline{r}'\|\right)$$



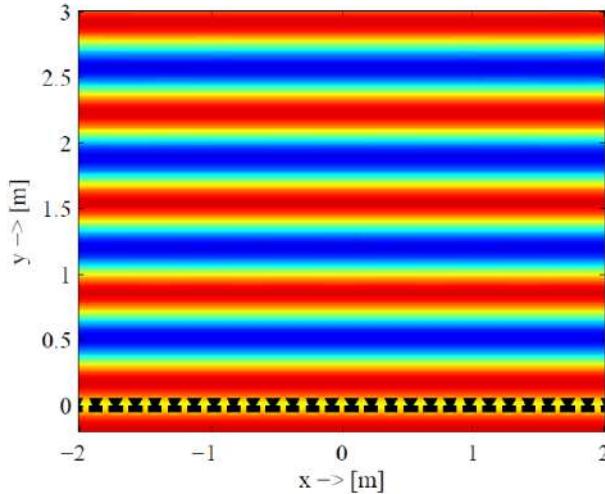
If we want to model the driving function (that is independent on the underlying dimensionality) producing a plane wave, the same formula we used for the 3D case holds:

$$D_{pw,2D}(\underline{r}', \omega) = D_{pw,3D}(\underline{r}', \omega) = 2a(\underline{r}') j\omega A(\omega) \frac{\langle \underline{k}_0, \hat{\underline{n}}(\underline{r}') \rangle}{c} e^{j\langle \underline{k}_0, \hat{\underline{n}}(\underline{r}') \rangle}$$

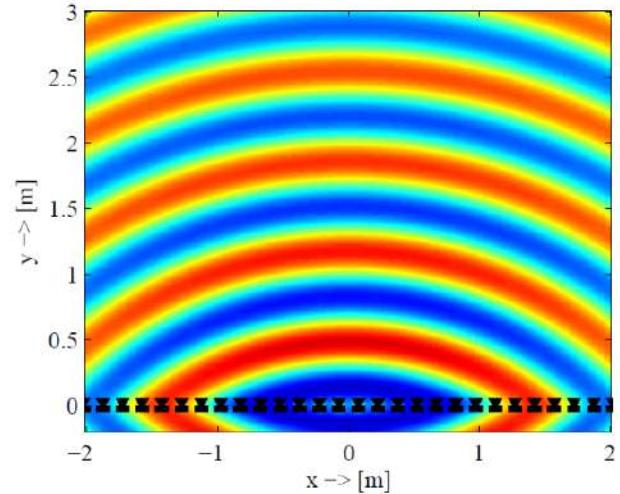
While for the case of a line source sound filed with source at  $\underline{z}$  is we need a driving function defined ad hoc:

$$D_{ls,2D}(\underline{r}', \omega) = \frac{1}{2c} a(\underline{r}') \frac{\langle \underline{r}' - \underline{z}, \hat{\underline{n}}(\underline{r}') \rangle}{\|\underline{r}' - \underline{z}\|} \times \frac{j\omega}{c} A(\omega) H_1^{(2)}\left(\frac{\omega}{c} \|\underline{r}' - \underline{z}\|\right)$$

Let us see a couple of examples:



(a) plane wave ( $f_{pw} = 500$  Hz,  $\mathbf{n}_{pw} = [0 \ 1]^T$ )



(b) cylindrical wave ( $f_{cy} = 500$  Hz,  $\mathbf{x}_S = [0 \ -2]^T$  m)

On the left, a virtual plane wave propagating along the  $y$  direction. On the right a cylindrical wave propagation model with decay in amplitude (due to the model itself).

### Derivation of wave field synthesis – 2.5D Wave field Synthesis

Actually, in practical scenarios, both the two decays are what we need, but it is difficult to realise them with real loudspeakers. Especially for linear sources, there are very few setups that well approximate them as a loudspeaker is generally more similar to a point source model. Therefore, we need to find a model that can be implemented with point sources as secondary sources.

In order to do it, let us recall the approximation (large argument approximation of the Hankel function) for the 2D Green's function

$$G_{2D}(\underline{r}'|\underline{r}, \omega) \approx \sqrt{\frac{2\pi \|\underline{r} - \underline{r}'\|}{j\omega}} G(\underline{r}'|\underline{r}, \omega)$$

Amplitude correction depends on the listening position: in the expression appears, in fact, a reference position  $\underline{r}_0$  used for amplitude correction. We call the scenario in which we have point sources that render a bidimensional sound field as 2.5D wave field synthesis. Once again let us see the driving function for the loudspeakers in this scenario. Its general expression is

$$D_{2.5D}(\underline{r}', \omega) = \underbrace{\sqrt{\frac{2\pi \|\underline{r}_0 - \underline{r}'\|}{j\omega}}}_{\text{Amplitude correction factor}} D_{3D}(\underline{r}', \omega)$$

Let us plug the expression of the 3D driving function in different scenarios in order to see how  $D_{2.5D}$  expression becomes.

Firstly, the driving function for a plane wave virtual field

$$\begin{aligned}
 D_{pw,2.5D}(\underline{r}', \omega) &= \sqrt{\frac{2\pi\|\underline{r}_0 - \underline{r}'\|}{\frac{j\omega}{c}}} D_{pw,3D}(\underline{r}', \omega) = \\
 &= \sqrt{\frac{2\pi\|\underline{r}_0 - \underline{r}'\|}{\frac{j\omega}{c}}} 2a(\underline{r}') j\omega A(\omega) \frac{\langle \hat{k}_0, \hat{n}(\underline{r}') \rangle}{c} e^{j\langle \underline{k}_0, \underline{r}' \rangle} = \\
 &= 2a(\underline{r}') \sqrt{2\pi\|\underline{r}_0 - \underline{r}'\|} \sqrt{j\frac{\omega}{c}} A(\omega) \langle \hat{k}_0, \hat{n}(\underline{r}') \rangle e^{j\langle \underline{k}_0, \underline{r}' \rangle}
 \end{aligned}$$

That, transformed in time domain becomes

$$d_{pw,2.5D}(\underline{r}', t) = w_{pw} \delta\left(t - \frac{\langle \underline{k}_0, \underline{r}' \rangle}{c}\right) * (f_{pw}(t) * A(t))$$

Where:

- $w_{pw}$  collects all the weighting terms. Explicitly, it is equal to

$$w_{pw} = 2a(\underline{r}') \sqrt{2\pi\|\underline{r}_0 - \underline{r}'\|} \langle \hat{k}_0, \hat{n}(\underline{r}') \rangle$$

- $f_{pw}(t)$  is simply the temporal inverse Fourier transform of the derivative term

$$f_{pw}(t) = \mathcal{F}_t^{-1} \left\{ \sqrt{j\frac{\omega}{c}} \right\}$$

Let us now see the scenario in which we want to model a point source virtual field. The driving function can be expressed by

$$\begin{aligned}
 D_{ps,2.5D}(\underline{r}', \omega) &= \sqrt{\frac{2\pi\|\underline{r}_0 - \underline{r}'\|}{\frac{j\omega}{c}}} D_{ps,3D}(\underline{r}', \omega) = \\
 &= \sqrt{\frac{2\pi\|\underline{r}_0 - \underline{r}'\|}{\frac{j\omega}{c}}} - 2a(\underline{r}') \frac{\langle \underline{r}' - \underline{z}, \hat{n}(\underline{r}') \rangle}{\|\underline{r}' - \underline{z}\|^2} \times \left( \frac{1}{\|\underline{r}' - \underline{z}\|} + \frac{j\omega}{c} \right) A(\omega) e^{-j\frac{\omega}{c}\|\underline{r}' - \underline{z}\|} = \\
 &= -2a(\underline{r}') \frac{\langle \underline{r}' - \underline{z}, \hat{n}(\underline{r}') \rangle}{\|\underline{r}' - \underline{z}\|^2} \sqrt{2\pi\|\underline{r}_0 - \underline{r}'\|} \times \left( \frac{1}{\sqrt{\frac{j\omega}{c}} \|\underline{r}' - \underline{z}\|} + \sqrt{\frac{j\omega}{c}} \right) A(\omega) e^{-j\frac{\omega}{c}\|\underline{r}' - \underline{z}\|}
 \end{aligned}$$

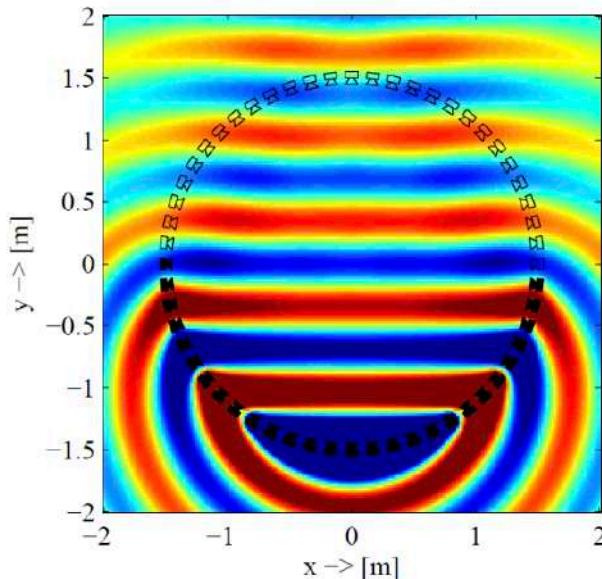
That, in time domain becomes

$$d_{ps,2.5D}(\underline{r}', t) = w_{ps} \delta\left(t - \frac{\|\underline{r}' - \underline{z}\|}{c}\right) * (f_{ps}(t) * A(t))$$

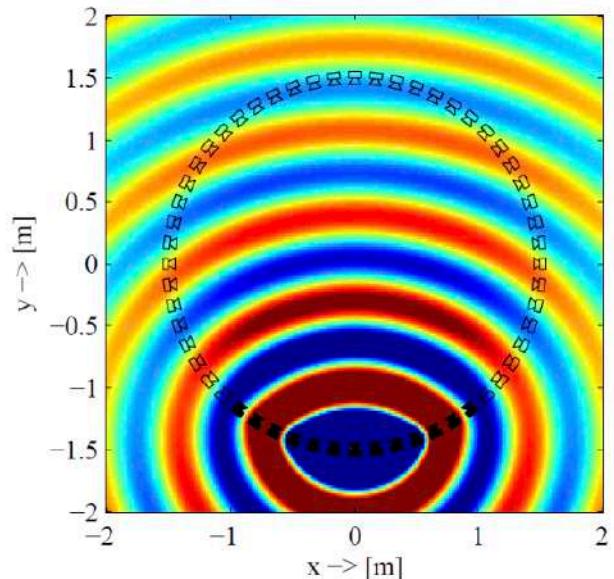
Here again:

- $w_{ps} = (\underline{r}') \frac{\langle \underline{r}' - \underline{z}, \hat{n}(\underline{r}') \rangle}{\|\underline{r}' - \underline{z}\|^2} \sqrt{2\pi \|\underline{r}_0 - \underline{r}'\|}$
- $f_{pw}(t) = \mathcal{F}_t^{-1} \left\{ \frac{1}{\sqrt{\frac{j\omega}{c}} \|\underline{r}' - \underline{z}\|} + \sqrt{\frac{j\omega}{c}} \right\}$

This is what we practically obtain



(a) plane wave ( $f_{pw} = 500$  Hz,  $\mathbf{n}_{pw} = [0 \ 1]^T$ )



(b) spherical wave ( $f_{sw} = 500$  Hz,  $\mathbf{x}_S = [0 \ -2]^T$  m)

The loudspeakers that are not filled in back are muted by the selection function. In the image on the right, where we want to render a line source that produces cylindrical waves, we can see that many of the loudspeakers are muted due to the selection function.

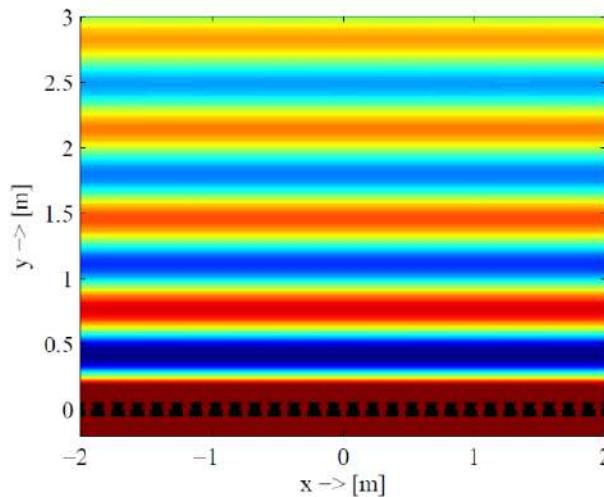
As we have done in the previous case, since the 2D contour  $\partial\mathcal{V}$  can degenerate into an infinite line, we integrate the product between the 2.5D driving function and the 3D Green's function along the  $x$  axis in order to obtain the overall pressure field representation

$$p(\underline{r}, \omega) = - \int_{-\infty}^{\infty} D_{2.5D}(\underline{r}', \omega) G_{3D}(\underline{r} | \underline{r}', \omega) dx'$$

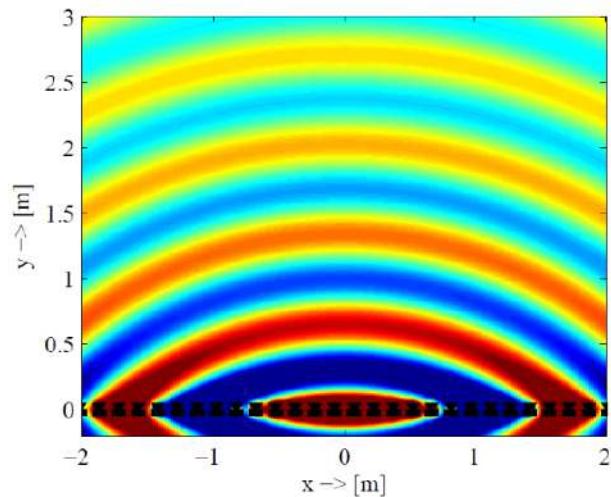
Let us see no which are the restrictions that we have to take into account to make the expression hold:

- Reproduced field will be symmetrical to  $x$  axis
- To have a plane wave the incident angle must be reduced to  $\pi < \phi_0 < 2\pi$
- To have point source the position is restricted to be below  $x$  axis
- This time, no explicit selection is required (there is no selection function that turns on or off the speakers)

Let us see this last example:



(a) plane wave ( $f_{\text{pw}} = 500$  Hz,  $\mathbf{n}_{\text{pw}} = [0 \ 1]^T$ )



(b) spherical wave ( $f_{\text{sw}} = 500$  Hz,  $\mathbf{x}_S = [0 \ -2]^T$  m)

In the first case we can see the plane waves case with the typical decay in amplitude, while on the right we can see the rendering of a sound field due to a cylindrical virtual source. Both generated by a linear distribution of point sources.

### Artefacts of wave field synthesis

As we said, in the actual implementation of wave field synthesis, some artefacts arise. There are mainly three of them:

#### 1. Spatial aliasing artefacts

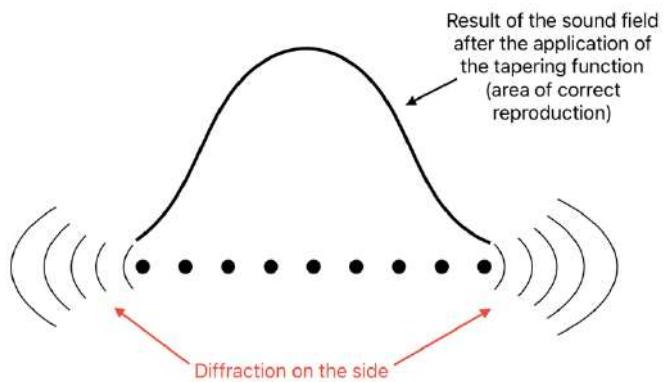
Spatial sampling of the continuous secondary source distribution leads to spatial aliasing artefacts.

The distortion of the spatial structure of the reproduced sound field that they introduce can make hard the human localisation of the virtual source and, in addition, can give the perception of timbral coloration. In order to stem their effects, we should conduct a rigorous analysis of them, but this is possible only when we consider a particular geometry for the secondary source distribution. What we do is to take into account two considerations that we exploit to reduce the effects. These are that the spatial aliasing increases with the bandwidth of the virtual source signal and that it depends on the listener position.

#### 2. Artefacts related to the truncation of secondary sources

Obviously, a practical implementation with open contours (i.e. linear arrays) will always be of finite size: we cannot physically realise an infinite array of loudspeaker. This truncation leads to artefacts as the area of exact reproduction becomes limited and diffraction arises from the outer secondary sources. To reduce this kind of artefacts we apply a **tapering window** (like the one reported in the image on the right) to the secondary source driving signal. The application of this window reduces the diffraction, but it further reduces the area of correct reproduction

<i>Tapering</i>	<i>Rastremazione</i>



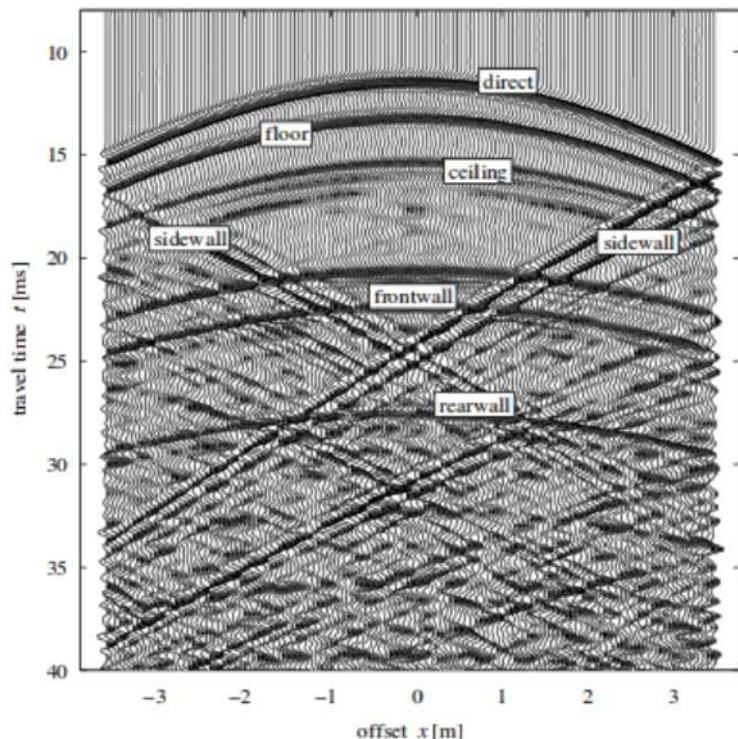
### 3. Amplitude errors

2D WFS systems employ point sources (loudspeakers) as secondary sources instead of dipoles and monopoles as it would be in the ideal case. This secondary source mismatch causes amplitude errors in the reproduced sound field. Furthermore, 2D WFS systems are designed to reproduce the desired field in a plane only. For this reason, artefacts arise for listeners that are not located in that plane: out of plane listeners have the impression that virtual sources are elevated or lowered.

## Wave Field Analysis

Due to the reciprocity principle, the Kirchhoff-Helmholtz integral can be used as a basis to build also a hypothetical wave field analysis system. In order to do this, we need to use both pressure and pressure gradient microphones on the boundary taking into account all the restriction seen for the reproduction system. Simply speaking, by knowing pressure values at the boundaries, we can characterise the pressure field inside the volume.

Let us see an example in which we record a multichannel signal with an array of microphones. As we can notice from the image, this set of recordings gives many insights in the temporal and spatial structure of the sound field. Reflection and diffraction can be easily discriminated: they are revealed by spatial correlation between neighbouring responses. An interesting aspect to notice is that wave field analysis is so informative that even the geometry of the room can be inferred. Anyway, in practice the array is composed by omnidirectional microphones. Therefore, no discrimination is possible in the elevation plane.



## Wave field Extrapolation

Responses at any other position in front of the array can be extrapolated based on Kirchhoff-Helmholtz integral. In this case we must apply some modifications to account only for pressure signals. They allow to compute the sound field at loudspeaker position, that can be used for reproduction.

## References

- [1] S. Spors, R. Rabenstein, and J. Ahrens. The theory of wave field synthesis revisited. In Proc. AES 124th Conv., Amsterdam, NE, May 17-20 2008.
- [2] A. J. Berkhout, D. D. Vries, and P. Vogel. Acoustic control by wave field synthesis. J. Acoust. Soc. Am., 93(5): 2764-2778, May 1993.
- [3] S. Spors and R. Rabenstein. Spatial aliasing artifacts produced by linear and circular loudspeaker arrays used for wave field synthesis. In Proc. AES 120th Conv., Paris, FR, May 20-23 2006.
- [4] S. Spors. Investigation of spatial aliasing artifacts of wave field synthesis in the temporal domain. In Proc. 34rd German Annual Conf. on Acoustics (DAGA), 2008.
- [5] D. de Vries and M. M. Boone. Wave field synthesis and analysis using array technology. In Proc. IEEE Workshop on Applications of Signal Process. to Audio and Acoustics (WASPAA), 1999.
- [6] F. Antonacci, J. Filos, M. R. P. Thomas, E. A. P. Habets, A. Sarti, P. A. Naylor, and S. Tubaro. Inference of room geometry from acoustic impulse responses. IEEE Transactions on Audio, Speech and Language Processing, 20 (10):2683-2695, Dec. 2012.
- [7] I. Dokmanić, R. Parhizkar, A. Walther, Y. M. Lu, and M. Vetterli. Acoustic echoes reveal room shape. PNAS, 110 (30):12186-12191, 2013

## Ambisonics

Today's class will be devoted to Ambisonics. This technique is based on the Fourier decomposition of a sound field.

The main purpose of arbitrary order Ambisonics methods (also known as “mode-matching” methods) is to extend panning methods by not using just the loudspeakers closest to the desired source direction but, instead, by using all the loudspeaker in the space we are considering.

As we said, the system exploit the fact that the sound field can be represented by a set of signals (coefficients  $c_n(\omega)$  of basis functions  $\beta(\underline{r}, \omega)$ ) that are:

- Independent from the setup of microphones used for recording
- Independent from the setup of loudspeakers used for reproduction

$$p(\underline{r}, \omega) = \sum_{n=-\infty}^{\infty} c_n(\omega) \beta(\underline{r}, \omega)$$

Let us have an overview on the outline of the lesson.

This time we will start from the 2D case where:

- Representation of the sound field is based on circular (cylindrical) harmonics
- Listener and loudspeakers are placed on the same plane
- Actuators that produce the sound field should be modelled as line sources that, under the far-field assumption, can be modelled also as plane wave sources.

As we said many times, a loudspeaker behaves more like a point source instead of a line one. This generates a model mismatch that we will have to fix.

At the end, we will move to the 3D case considering spherical harmonics.

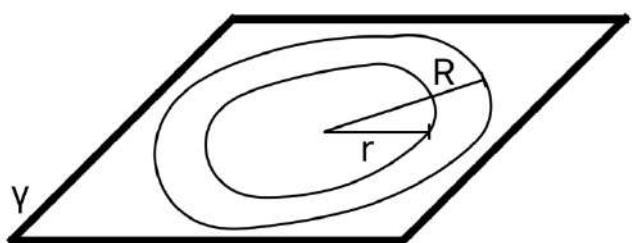
### Planar Geometry (2D Case)

We start from a condition of planar geometry (2D environment). Arbitrary order 2D ambisonics methods can be applied using whichever planar microphone array geometry in the analysis phase and whichever planar loudspeaker array geometry in the rendering phase.

In this case, the microphone array geometry can be different to the loudspeaker array geometry as the ambisonics is based the decomposition of the signal on some basic functions (independently from the system used to acquire the signal and to the rendering one).

For the sake of simplicity, but without loss of generality, we make the following assumptions

- We consider a distribution of loudspeakers and a listener on the  $xy$  plane
- We assume that the distribution of loudspeakers is placed on a circle with radius  $R$  on the  $xy$  plane
- We are interested in representing the sound field in a circular area (defined as the listening area) inside the circle of loudspeakers where the radial coordinate of points inside the listening area satisfies  $r < R$ .



The setup is like the one reported on the right. The listener is assumed to be in the area of radius  $r$ .

Before going on, let us do a brief recap on the representation of the sound field. We remind that we call this scenario internal sound field as we are solving an internal problem (we want to model the sound field enclosed). The internal sound field in terms of circular harmonics can be written as a linear combination of spherical harmonics

$$p(\underline{r}, \omega) = \sum_{m=-\infty}^{\infty} C_m(\omega) J_m \left( \frac{\omega}{c} r \right) e^{jm\phi}$$

Since it is not possible to perform infinite summation, we truncate the sum to  $|m| \leq M$

$$p_M(\underline{r}, \omega) = \sum_{m=-M}^M C_m(\omega) J_m \left( \frac{\omega}{c} r \right) e^{jm\phi}$$

We remind that just the Bessel functions of order zero is equal to one when the argument is equal to zero. Furthermore, if we consider low arguments (therefore, if we are close to the origin in the graph on the right), we notice that the lower order Bessel functions are the one that give more contribution. We notice in fact that  $J_1(x)$  reaches his peaks well before than  $J_0(x)$ . Another thing that we highlight is the fact that there are several zero-crossing: this becomes a problem when the function is placed at the denominator.

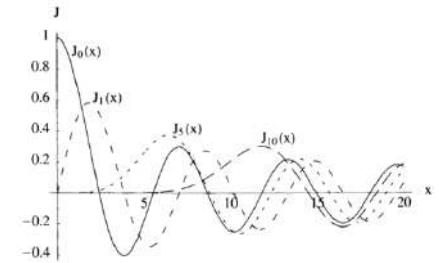


Figure: Bessel functions of the first kind for  $m = 0, 1, 5, 10$ .  
Figure taken from [1, p. 119].

We said that we have to truncate the sum. But, how much information is lost by performing this truncation? Let us define a truncation error

$$\epsilon_M(\underline{r}) = \frac{|p(\underline{r}, \omega) - p_M(\underline{r}, \omega)|}{2\pi |p(\underline{0}, \omega)|}$$

We know that the truncation error is bounded by a value equal to  $\eta e^{-\delta}$  where  $\eta \approx 0.16127$  and  $\delta \in \mathbb{N}$  if the condition  $M = \left\lceil \frac{er(\frac{\omega}{c})}{2} \right\rceil$  is satisfied. This formula is telling us that under this condition, the error is small and acceptable in most situation (no more than 16% once  $M$  equals the critical threshold). If we want to increase our listening area ( $r$ ), we have to increase also the order of expansion  $M$ . Also, if we want to increase the frequency  $\omega$ , we have to increase also the order of the expansion  $M$ . In practice, depending on the application, the listening area have a certain radius and the signal a maximal frequency  $\omega_{max}$ . As we know this parametres we look for the truncation order  $\bar{M}$  as

$$\bar{M} = \left\lceil e \frac{\omega_{max}}{2c} r_{max} \right\rceil$$

So, this truncation order  $\bar{M}$  will be sufficient to have a truncation error less than 16% for all frequencies  $\omega < \omega_{max}$  and for all the points with distance  $r < r_{max}$ .

However, often the series is truncated to a low order  $M$ . The case in which  $M = 1$  (truncation to the first order) is often referred to as Ambisonics method. In this case, the sound field is fully characterised by three coefficients  $C_0(\omega), C_1(\omega), C_{-1}(\omega)$  where, for simplicity, the last two are chosen such that  $C_1(\omega) = C_{-1}(\omega)$ . This results in a model of the sound field that is symmetrical with respect to a line on the  $xy$  plane. With this setting, coefficients can be interpreted as the weights of a pressure component ( $C_0(\omega)$ ,

monopole) and two orthogonal particle velocity components ( $C_1(\omega)$  and  $C_{-1}(\omega)$ ) two dipoles, one on the x-axis and one on the y-axis). The cases in which  $M > 1$  are often referred to as Higher-Order Ambisonics (HOA) methods.

### ***First-Order ambisonics***

Which are the advantages of having First-Order ambisonics?

- Low complexity
- Few loudspeakers

The size of the listening area (sweet spot) enlarges with the number of loudspeakers. What happens in practice is that in the sweet spot the auditory event is aligned with the DOA of the virtual source while, outside the sweet spot, auditory scene often collapses towards the closest loudspeaker.

### ***Higher-Order ambisonics***

For higher order ambisonics, the characteristics are the following:

- Higher complexity
- The number of needed loudspeakers increases with the order (see details later)

In this case, the size of the listening area (sweet spot) enlarges with the number of loudspeakers, but also enlarges with the order.

Our aim is to find the coefficients  $C_m(\omega)$  such that a target desired sound field  $p_{target}(\underline{r}, \omega)$  can be approximated as

$$p_{target}(\underline{r}, \omega) = \sum_{m=-M}^{M} C_m(\omega) J_m \left( \frac{\omega}{c} r \right) e^{jm\phi}$$

We have to consider two different scenarios

- In a model-based scenario,  $C_m(\omega)$  depend on the model of the desired sound field
- In a data-based scenario,  $C_m(\omega)$  are inferred from the recorded microphone signals. In this case, we will not be able to find a close form expression

Let us start from the simplest case of model-based scenario:

#### ***Plane wave source***

Let us formalise the problem: we want to reproduce the sound field of a plane wave with unitary amplitude and desired direction of arrival  $\underline{k}_0$

$$p_{target}(\underline{r}, \omega) = e^{j\langle \underline{k}_0, \underline{r} \rangle}$$

Where  $\underline{k}_0 = \frac{\omega}{c} \begin{pmatrix} \cos(\phi_0) \\ \sin(\phi_0) \end{pmatrix}^T$  is the wavenumber vector (in which  $\phi_0$  is the DOA of the soundwave). In polar coordinates, the inner product becomes

$$\langle \underline{k}_0, \underline{r} \rangle = \frac{\omega}{c} r \cos(\phi_0 - \phi)$$

In light of this, we can re-write the  $p_{target}(\underline{r}, \omega)$  signal as it follows noticing that the exponential is a periodic function of  $\phi$

$$p_{target}(\underline{r}, \omega) = e^{j\frac{\omega}{c}r \cos(\phi_0 - \phi)}$$

Since it is a periodic function, it can be expressed with the Fourier series

$$e^{j\frac{\omega}{c}r \cos(\phi_0 - \phi)} = \sum_{m=-\infty}^{\infty} B_m(\omega) e^{jm\phi}$$

Now that we have the sound field emanated by the source and our sound field, we put the one equal to the other exploiting the Jacobi–Anger expansion

#### Reminder: Jacobi–Anger expansion

In mathematics, the Jacobi–Anger expansion (or Jacobi–Anger identity) is an expansion of exponentials of trigonometric functions in the basis of their harmonics. It is useful in physics (for example, to convert between plane waves and cylindrical waves), and in signal processing (to describe FM signals). This identity is named after the 19th-century mathematicians Carl Jacobi and Carl Theodor Anger. Jacobi is similar to a Disney's character (a frog, if I am not wrong) but I cannot remember who...



$$e^{jz \cos(\phi)} = \sum_{m=-\infty}^{\infty} j^m J_m(z) \cdot e^{jm\phi}$$

The target sound field can be exactly written in the same shape

$$p(\underline{r}, \omega) = p_{target}(\underline{r}, \omega) = e^{j\frac{\omega}{c}r \cos(\phi - \phi_0)}$$

By applying the cylindrical harmonic expansion to the signal  $p(\underline{r}, \omega) = \sum_{m=-\infty}^{\infty} C_m(\omega) J_m\left(\frac{\omega}{c}r\right) e^{jm\phi}$  and the Jacobi–Anger expansion to the target signal  $p_{target}(\underline{r}, \omega) = \sum_{m=-\infty}^{\infty} j^m J_m\left(\frac{\omega}{c}r\right) \cdot e^{jm(\phi - \phi_0)}$ , we get that

$$\sum_{m=-\infty}^{\infty} C_m(\omega) J_m\left(\frac{\omega}{c}r\right) e^{jm\phi} = \sum_{m=-\infty}^{\infty} j^m J_m\left(\frac{\omega}{c}r\right) \cdot e^{jm(\phi - \phi_0)}$$

By simplifying the common terms, we end up with

$$C_m(\omega) = j^m \cdot e^{-jm\phi_0}$$

We call the model **mode matching** since we are evaluating the coefficients  $C_m(\omega)$  for all the values of  $m$  (so, for every mode).

### ***Line source***

Let us now consider a line source that is outside the listening area. The sound generated by a line source at point  $\underline{z}$  with coordinates  $(z, \phi_z)$  (with  $z > r$ ) can be expressed by

$$p_{target}(\underline{r}, \omega) = \frac{j}{4} H_0^{(2)}\left(\frac{\omega}{c} \|\underline{r} - \underline{z}\|\right)$$

By applying the well-known (mah insomma) addition property for the Hankel function, we get that

$$p_{target}(\underline{r}, \omega) = \frac{j}{4} H_0^{(2)}\left(\frac{\omega}{c} \|\underline{r} - \underline{z}\|\right) \underset{\substack{\text{Addition for} \\ \text{the Hankel function}}}{=} \frac{j}{4} \sum_{m=-\infty}^{\infty} H_m^{(2)}\left(\frac{\omega}{c} z\right) e^{-jm\phi_z} J_m\left(\frac{\omega}{c} r\right) e^{jm\phi}$$

Once again, we can find the desired coefficients by matching mode by mode

$$p(\underline{r}, \omega) = \sum_{m=-\infty}^{\infty} C_m(\omega) J_m\left(\frac{\omega}{c} r\right) e^{jm\phi} = \frac{j}{4} \sum_{m=-\infty}^{\infty} H_m^{(2)}\left(\frac{\omega}{c} z\right) e^{-jm\phi_z} J_m\left(\frac{\omega}{c} r\right) e^{jm\phi} = p_{target}(\underline{r}, \omega)$$

By simplifying the common terms, we get that

$$C_m(\omega) = \frac{j}{4} H_m^{(2)}\left(\frac{\omega}{c} z\right) e^{-jm\phi_z}$$

### ***Data-based scenario***

This is what happens in practical scenarios where the model of the source is not available, or it is too complex to be used. During the analysis phase, the coefficients  $C_m(\omega)$  can be estimated using the signals recorded with microphones. The same weight coefficients  $C_m(\omega)$  can then be used during the rendering for reproducing the desired sound field.

## Rendering Approaches

We now want to move to how to reproduce the sound field. Formally, we want to find the proper weights to apply to the driving signals of loudspeakers in order to reproduce the desired sound field.

The problem is approached by means of two different techniques:

- Model-based rendering  
where it is assumed that the source is characterised by a specific model (plane wave, line source) and, depending on whether or not we are making the far-field assumption, the loudspeakers are modelled as, respectively, plane waves or line sources. The model of secondary sources (loudspeakers) can be different to the model of the desired source.
- Data-based rendering  
where the source model is not assumed at all. In this context,  $C_m(\omega)$  coefficients are estimated from the microphone signals recorded in the analysis phase. Specific models of the secondary sources (loudspeakers) can still be used to derive the loudspeaker weights.

### *Loudspeakers modelled as plane waves*

Let us start from the scenario in which loudspeakers are modelled as plane waves. This scenario is valid under the following assumptions:

- Far field assumption: loudspeakers are far enough from the listener so that their field can be modelled as a plane wave. If we cannot rely on this assumption, we have a model mismatch and we have to use some approximations.
- The sound field in the  $xy$  plane can be approximated by a superposition of plane waves (it is valid as long as we are in a linear acoustic scenario where the superposing of effects principle holds).

The process is still the same: we have to find a way to make  $p(\underline{r}, \omega)$  match  $p_{target}(\underline{r}, \omega)$ . So, the sound field resulting from  $Q$  loudspeakers (in plane wave assumption) can be modelled by

$$p(\underline{r}, \omega) = \sum_{q=0}^{Q-1} d_q(\omega) e^{j\langle \underline{k}_q, \underline{r} \rangle}$$

What we have to do is to find loudspeaker coefficients  $d_q(\omega)$  such that

$$p_{target}(\underline{r}, \omega) = \sum_{q=0}^{Q-1} d_q(\omega) e^{j\langle \underline{k}_q, \underline{r} \rangle}$$

And how can we find them? By solving the linear system of equations described by the last expression. Let us try to analyse an example in which the model-based rendering is implemented by loudspeakers that reproduce plane waves generating a plane wave sound field. Sound field that can be described by

$$p_{target}(\underline{r}, \omega) = e^{j\langle \underline{k}_0, \underline{r} \rangle}$$

In this condition, our aim becomes finding the coefficients  $d_q(\omega)$  such that

$$p_{target}(\underline{r}, \omega) = e^{j\langle \underline{k}_0, \underline{r} \rangle} = \sum_{q=0}^{Q-1} d_q(\omega) e^{j\langle \underline{k}_q, \underline{r} \rangle}$$

Similar to what we did before, we exploit the Jacobi-Anger expansion, this time applied on both sides

$$\sum_{m=-\infty}^{\infty} j^m J_m \left( \frac{\omega}{c} r \right) \cdot e^{jm(\phi - \phi_0)} = \sum_{q=0}^{Q-1} d_q(\omega) \sum_{m=-\infty}^{\infty} j^m J_m \left( \frac{\omega}{c} r \right) \cdot e^{jm(\phi - \phi_q)}$$

If we exchange the summation order in this equation, we are able to use a modal matching approach finding the  $Q$  coefficients  $d_q(\omega)$ .

$$\sum_{m=-\infty}^{\infty} j^m J_m \left( \frac{\omega}{c} r \right) \cdot e^{jm(\phi - \phi_0)} = \sum_{m=-\infty}^{\infty} \sum_{q=0}^{Q-1} d_q(\omega) j^m J_m \left( \frac{\omega}{c} r \right) \cdot e^{jm(\phi - \phi_q)}$$

Limiting ourselves in a scenario of uniform circular array (in which  $\phi_q = q \frac{2\pi}{Q}$ ) and obviously simplifying the commons terms, we get that

$$\begin{aligned} \sum_{m=-\infty}^{\infty} j^m J_m \left( \frac{\omega}{c} r \right) \cdot e^{jm(\phi - \phi_0)} &= \sum_{m=-\infty}^{\infty} j^m J_m \left( \frac{\omega}{c} r \right) \sum_{q=0}^{Q-1} d_q(\omega) \cdot e^{jm(\phi - \phi_q)} \\ e^{jm\phi} e^{jm\phi_0} &= \sum_{q=0}^{Q-1} d_q(\omega) e^{jm\phi} e^{jm\phi_q} \\ e^{jm\phi_0} &= \sum_{q=0}^{Q-1} d_q(\omega) e^{jm\phi_q} \underset{\phi_q = q \frac{2\pi}{Q}}{\underset{\omega}{\equiv}} \sum_{q=0}^{Q-1} d_q(\omega) e^{jmq \frac{2\pi}{Q}} \triangleq DFT_Q \{d_q(\omega)\} \end{aligned}$$

The last expression leads exactly to the definition of the DFT on  $Q$  samples of  $d_q(\omega)$ . In order to find the coefficients, we apply the inverse DFT obtaining a nice and close expression.

$$d_q(\omega) = DFT_Q^{-1} \{e^{-jm\phi_0}\} = \frac{\sin \left( Q \frac{(\phi_0 - \phi_q)}{2} \right)}{Q \sin \left( \frac{(\phi_0 - \phi_q)}{2} \right)}$$

Once we found the lodusapekar weights, we are ready to plug them into the expression of the angular coefficients  $C_m(\omega)$

$$C_m(\omega) = \sum_{q=0}^{Q-1} d_q(\omega) j^m e^{-jm\phi_q}$$

Let us see this in practice in the following code

```

%% Ambisonics
% Lucio Bianchi - 25/05/2015

clear
close all
clc

% parameters
f = 800;
w = 2*pi*f;
c = 340;

% loudspeaker distribution
N = 7;
phi_n = 0:2*pi/N:2*pi-2*pi/N;
r_n = 2;

% listening area
Nx = 201;
Ny = 201;

x = linspace(-0.5,0.5,Nx);
y = linspace(-0.5,0.5,Ny);

[X, Y] = meshgrid(x, y);

% desired sound field
phi_0 = pi/3;

p_target = exp(1i*w/c*(cos(phi_0)*X + sin(phi_0)*Y));

figure
imagesc(x,y,real(p_target)), axis equal, axis tight, axis xy
title('Desired field'), xlabel('x [m]'), ylabel('y [m]')

% loudspeaker panning function
d_n = sin(N*(repmat(phi_0,1,N)-phi_n)/2) ./ ...
    (N * sin((repmat(phi_0,1,N)-phi_n)/2));

%% obtained sound field (assuming PW propagation)
p = zeros(Nx,Ny);
for n = 1:N
    p = p + d_n(n) * exp(1i*w/c*(cos(phi_n(n))*X + sin(phi_n(n))*Y));
end

figure
imagesc(x,y,real(p)), axis equal, axis tight, axis xy
title('Rendered field (PW sources)'), xlabel('x [m]'), ylabel('y [m]')

% error
err = abs(p_target.^2 - p.^2) ./ p_target.^2;

figure
imagesc(x,y,db(err)), colorbar, axis equal, axis tight, axis xy
title('Error (PW sources)'), xlabel('x [m]'), ylabel('y [m]')

%% obtained sound field (point sources)
p_hat = zeros(Nx,Ny);
for n = 1:N
    R = sqrt((X-r_n*cos(phi_n(n))).^2+(Y-r_n*sin(phi_n(n))).^2);

```

```

p_hat = p_hat + d_n(n) * exp(-1i*w/c*R) ./ (4*pi*R);
end

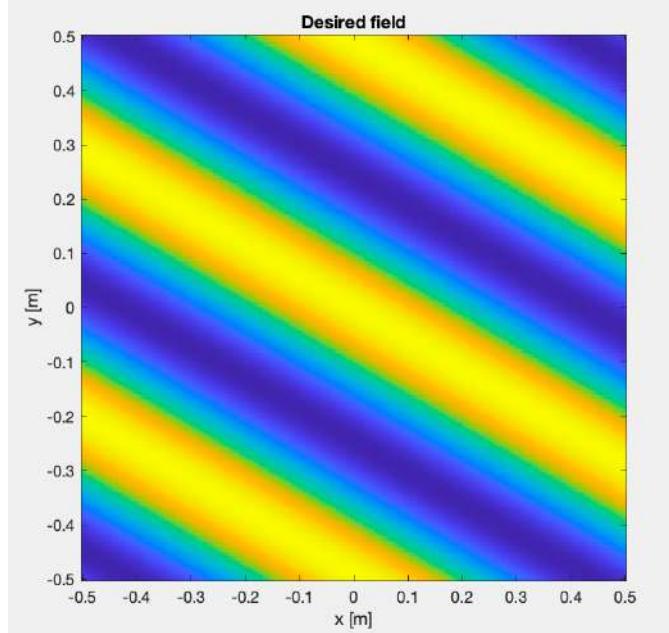
figure
imagesc(x,y,real(p_hat)), axis equal, axis tight, axis xy
title('Rendered field (point sources)'), xlabel('x [m]'), ylabel('y [m]')

% error
err_ps = abs(p_target.^2 - p_hat.^2) ./ p_target.^2;

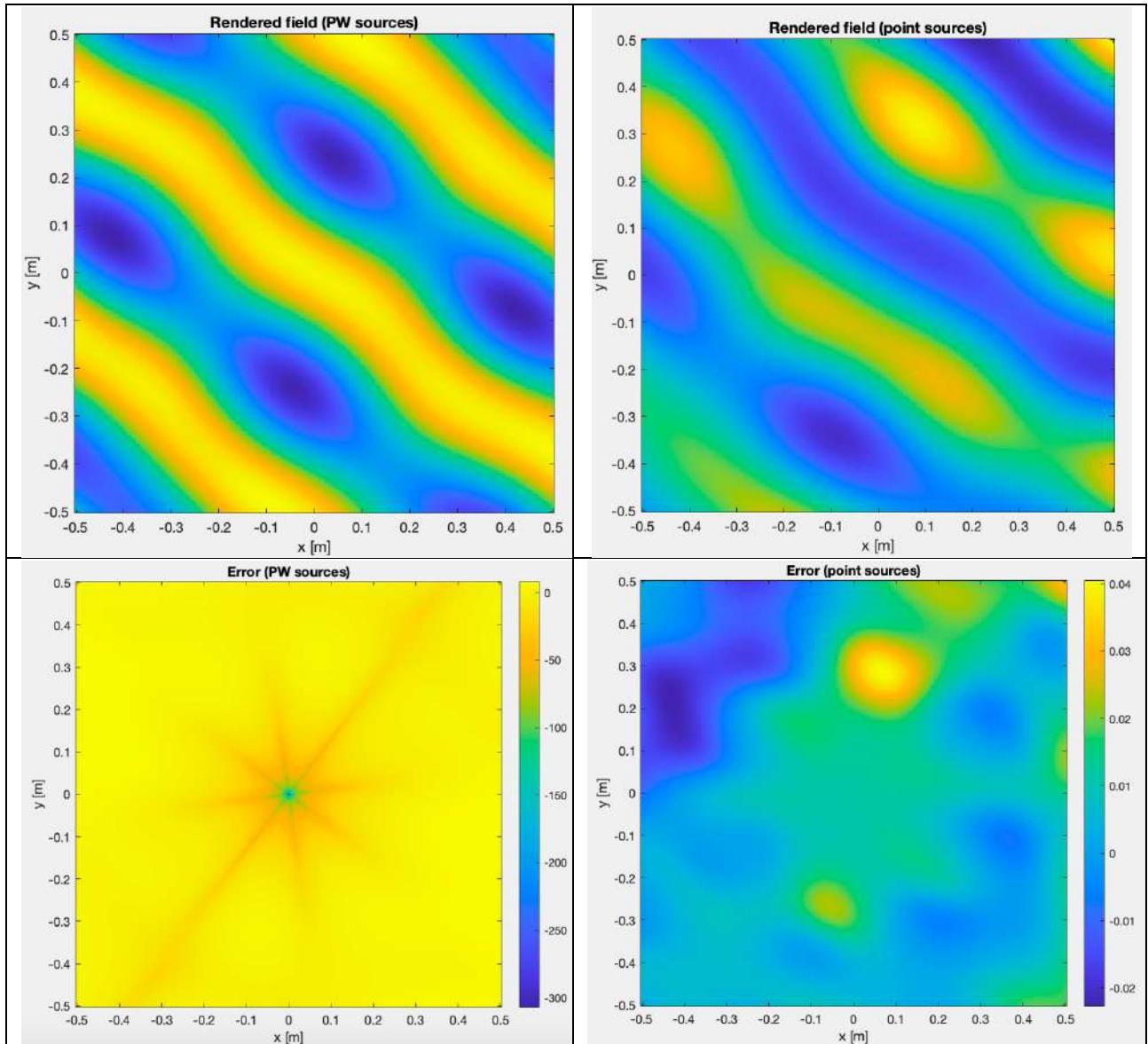
figure
imagesc(x,y,db(err_ps)), colorbar, axis equal, axis tight, axis xy
title('Error (point sources)'), xlabel('x [m]'), ylabel('y [m]')

```

This is the desired field that we want to reproduce



In the next page its rendering made with plane wave sources and point sources and their relative error is reported.



### Loudspeakers modelled as line sources

Let us see now how things changes if we consider a continuous loudspeaker distribution on a circle of radius  $R$ , i.e. ideal line sources are placed at  $\underline{v} = (R, \phi_v)$ . We set  $D(\phi_v, \omega)$  to be the driving function for the continuous loudspeaker distribution. Here, loudspeakers are assumed to be line sources orthogonal to the  $xy$  plane. Reproduced sound field is given by simple source formulation (for  $r < R$ )

$$p(\underline{r}, \omega) = \int_0^{2\pi} D(\phi_v, \omega) \frac{j}{4} H_0^{(2)} \left( \frac{\omega}{c} \|\underline{r} - \underline{v}\| \right) d\phi_v$$

Once again, we use a model match approach. We priorly need to introduce the addition theorem for cylindrical harmonics

$$H_0^{(2)} \left( \frac{\omega}{c} \|\underline{r} - \underline{v}\| \right) = \sum_{m=-\infty}^{\infty} H_m^{(2)} \left( \frac{\omega}{c} R \right) e^{-jm\phi_v} J_m \left( \frac{\omega}{c} r \right) e^{jm\phi}$$

In a real scenario,  $D(\phi_v, \omega)$  is a driving function of period  $2\pi$ , hence it can be expanded into a Fourier series

$$D(\phi_v, \omega) = \sum_{m=-\infty}^{\infty} B_m(\omega) e^{jm\phi_v}$$

Let us now rewrite the expression for the reproduced sound field accordingly to what we have just written.

$$\begin{aligned} p(\underline{r}, \omega) &= \int_0^{2\pi} D(\phi_v, \omega) \frac{j}{4} H_0^{(2)} \left( \frac{\omega}{c} \|\underline{r} - \underline{v}\| \right) d\phi_v \\ &\quad \stackrel{\text{Addition theorem for}}{=} \text{Henkel's functions for cylindrical harmonics} \\ &= \int_0^{2\pi} D(\phi_v, \omega) \frac{j}{4} \sum_{m=-\infty}^{\infty} H_m^{(2)} \left( \frac{\omega}{c} R \right) e^{-jm\phi_v} J_m \left( \frac{\omega}{c} r \right) e^{jm\phi} d\phi_v = \\ &= \int_0^{2\pi} \sum_{m=-\infty}^{\infty} B_m(\omega) e^{jm\phi_v} \frac{j}{4} \sum_{m=-\infty}^{\infty} H_m^{(2)} \left( \frac{\omega}{c} R \right) e^{-jm\phi_v} J_m \left( \frac{\omega}{c} r \right) e^{jm\phi} d\phi_v = \\ &= \sum_{m=-\infty}^{\infty} B_m \frac{j}{4}(\omega) H_m^{(2)} \left( \frac{\omega}{c} R \right) J_m \left( \frac{\omega}{c} r \right) e^{jm\phi} \int_0^{2\pi} e^{-jm\phi_v} e^{jm\phi_v} d\phi_v = \\ &= \sum_{m=-\infty}^{\infty} B_m \frac{j}{4}(\omega) H_m^{(2)} \left( \frac{\omega}{c} R \right) J_m \left( \frac{\omega}{c} r \right) e^{jm\phi} 1|_0^{2\pi} = \\ &= \sum_{m=-\infty}^{\infty} B_m \underbrace{\frac{j}{2} \pi(\omega) H_m^{(2)} \left( \frac{\omega}{c} R \right)}_{c_m(\omega)} J_m \left( \frac{\omega}{c} r \right) e^{jm\phi} \end{aligned}$$

As we can notice, the reproduced sound field is a weighted sum of basis functions

$$J_m\left(\frac{\omega}{c}r\right) e^{jm\phi}$$

$m$ -th mode weight is  $C_m(\omega) = B_m \frac{j}{2} \pi(\omega) H_m^{(2)}\left(\frac{\omega}{c} R\right)$ . By inverting the function, we are able to find the relationship between the coefficients  $C_m(\omega)$  of the cylindrical harmonics expansion with  $B_m(\omega)$

$$B_m(\omega) = \frac{2}{j\pi H_m^{(2)}\left(\left(\frac{\omega}{c}\right)r\right)} C_m(\omega) \quad m \in -M, \dots, M$$

So, the driving function will be nothing but

$$D(\phi_v, \omega) = \sum_{m=-M}^M \frac{2}{j\pi H_m^{(2)}\left(\left(\frac{\omega}{c}\right)r\right)} C_m(\omega) e^{jm\phi_v}$$

In real application we have to discretise the distribution as it is done the draw of the setup we are considering.  $D(\phi_v, \omega)$  is a periodic function of period  $2\pi$  whose maximum frequency is  $\frac{M}{2\pi}$ . We are doing a discretisation, according to the Shannon theorem we need  $2M + 1$  equidistant samples on the circle to accurately reproduce  $D(\phi_v, \omega)$ . If we consider  $Q$  loudspeakers ( $Q \geq 2M + 1$ ), the  $q$ th weight will be express by

$$d_q(\omega) = D(\phi_q, \omega) \Delta\phi \quad \text{where } \Delta\phi = \frac{2\pi}{Q} \text{ and } \phi_q = q\Delta\phi$$

The loudspeaker weights can be written as

$$d_q(\omega) = \sum_{m=-M}^M \frac{2}{j\pi H_m^{(2)}\left(\left(\frac{\omega}{c}\right)r\right)} C_m(\omega) e^{jm\phi_q} \Delta\phi \quad \text{with } q = 1, \dots, Q$$

That creates a relation between the ambisonics coefficients and the one of the loudspeakers.

Final remarks:

- Driving function depends on desired field coefficients, loudspeaker positions and frequency;
- No errors are introduced at low order harmonics ( $m = M, \dots, M$ );
- The higher the upper frequency  $\omega_{max}$ , the higher the number of needed loudspeakers  $Q$ ;
- In the design of the ambisonics system the relationship between the number of loudspeakers, the size of the reproduction region, the frequency range and the desired accuracy needs to be taken into account.

On Matlab

```
%% Higher-Order Ambisonics
% Lucio Bianchi - 25/05/2015

clear
close all
```

```

clc

% parameters
f = 800;
w = 2*pi*f;
c = 340;

% listening area
Nx = 201;
Ny = 201;

r_max = 1;

x = linspace(-2,2,Nx);
y = linspace(-2,2,Ny);

[X, Y] = meshgrid(x, y);

R = sqrt(X.^2 + Y.^2);

% compute the truncation order
M = ceil(exp(1)*r_max*(w/c)/2);
m = (-M:M)';

% loudspeaker distribution
Q = 2*M+1;
phi_v = (0:2*pi/Q:2*pi-2*pi/Q)';
v = 2;

% desired field: plane wave
phi_0 = pi/6;
p_pw = exp(1i*w/c*(cos(phi_0)*X + sin(phi_0)*Y));

figure
imagesc(x,y,real(p_pw)), colorbar, axis equal, axis tight, axis xy
title('Desired field PW'), xlabel('x [m]'), ylabel('y [m]')

% desired field: line source
z = 2;
phi_z = pi/3;
z_vec = z * [cos(phi_z); sin(phi_z)];

p_ls = 1i/4 * besselh(0,1,w/c*sqrt((z_vec(1)-X).^2 + (z_vec(2)-Y).^2));

figure
imagesc(x,y,real(p_ls)), colorbar, axis equal, axis tight, axis xy
title('Desired field LS'), xlabel('x [m]'), ylabel('y [m]')

% sound field coefficients
[MM, MPH1_v] = meshgrid(m, phi_v);
A_pw = 1i.^MM .* exp(-1i*MM*phi_0);
A_ls = 1i/4 * besselh(MM,1,w/c*z) .* exp(-1i*MM*phi_z);

% loudspeaker weights
D_pw = sum( 2 ./ (1i*pi*besselh(MM,1,w/c*v)) .* A_pw .* exp(1i*MM.*MPH1_v),
2);
D_ls = sum( 2 ./ (1i*pi*besselh(MM,1,w/c*v)) .* A_ls .* exp(1i*MM.*MPH1_v),
2);

% simulate sound fields
[PHI_v, XX] = meshgrid(phi_v, X(:));

```

```
[~, YY] = meshgrid(phi_v, Y(:));
G = 1i/4 * besselh(0,1,w/c*sqrt((XX-v*cos(PHI_v)).^2 + (YY-v*sin(PHI_v)).^2));

p_hat_pw = 2*pi/Q * G * D_pw;
p_hat_ls = 2*pi/Q * G * D_ls;

figure
imagesc(x,y,real(reshape(p_hat_pw, Nx, Ny))), colorbar, axis equal, axis tight, axis xy
title('Reproduced field PW'), xlabel('x [m]'), ylabel('y [m]')

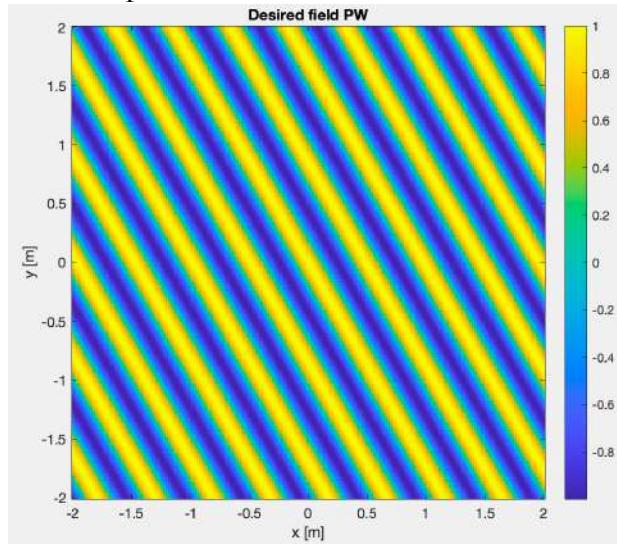
figure
imagesc(x,y,real(reshape(p_hat_ls, Nx, Ny))), colorbar, axis equal, axis tight, axis xy
title('Reproduced field LS'), xlabel('x [m]'), ylabel('y [m]')

% reproduction error
err_pw = abs(reshape(p_hat_pw, Nx, Ny) - p_pw) ./ ...
abs(2*pi*p_pw((Nx+1)/2,(Ny+1)/2));
err_ls = abs(reshape(p_hat_ls, Nx, Ny) - p_ls) ./ ...
abs(2*pi*p_ls((Nx+1)/2,(Ny+1)/2));

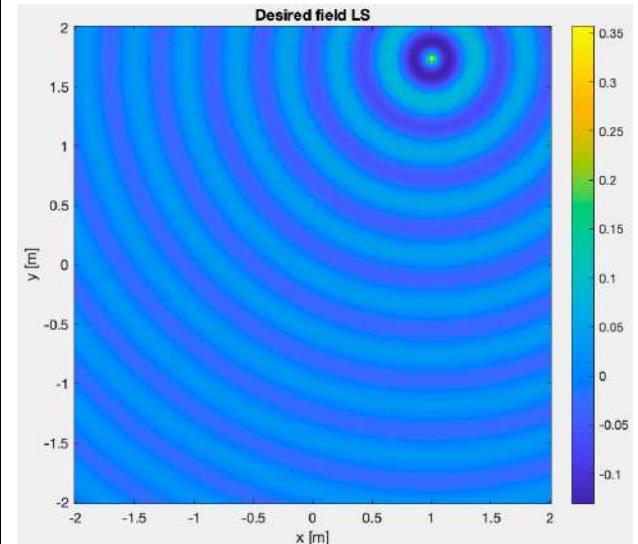
figure
imagesc(x,y,err_pw,[0.16 1]), colorbar, axis equal, axis tight, axis xy
title('Reproduction error PW'), xlabel('x [m]'), ylabel('y [m]')

figure
imagesc(x,y,err_ls, [0.16, 1]), colorbar, axis equal, axis tight, axis xy
title('Reproduction error LS'), xlabel('x [m]'), ylabel('y [m]')
```

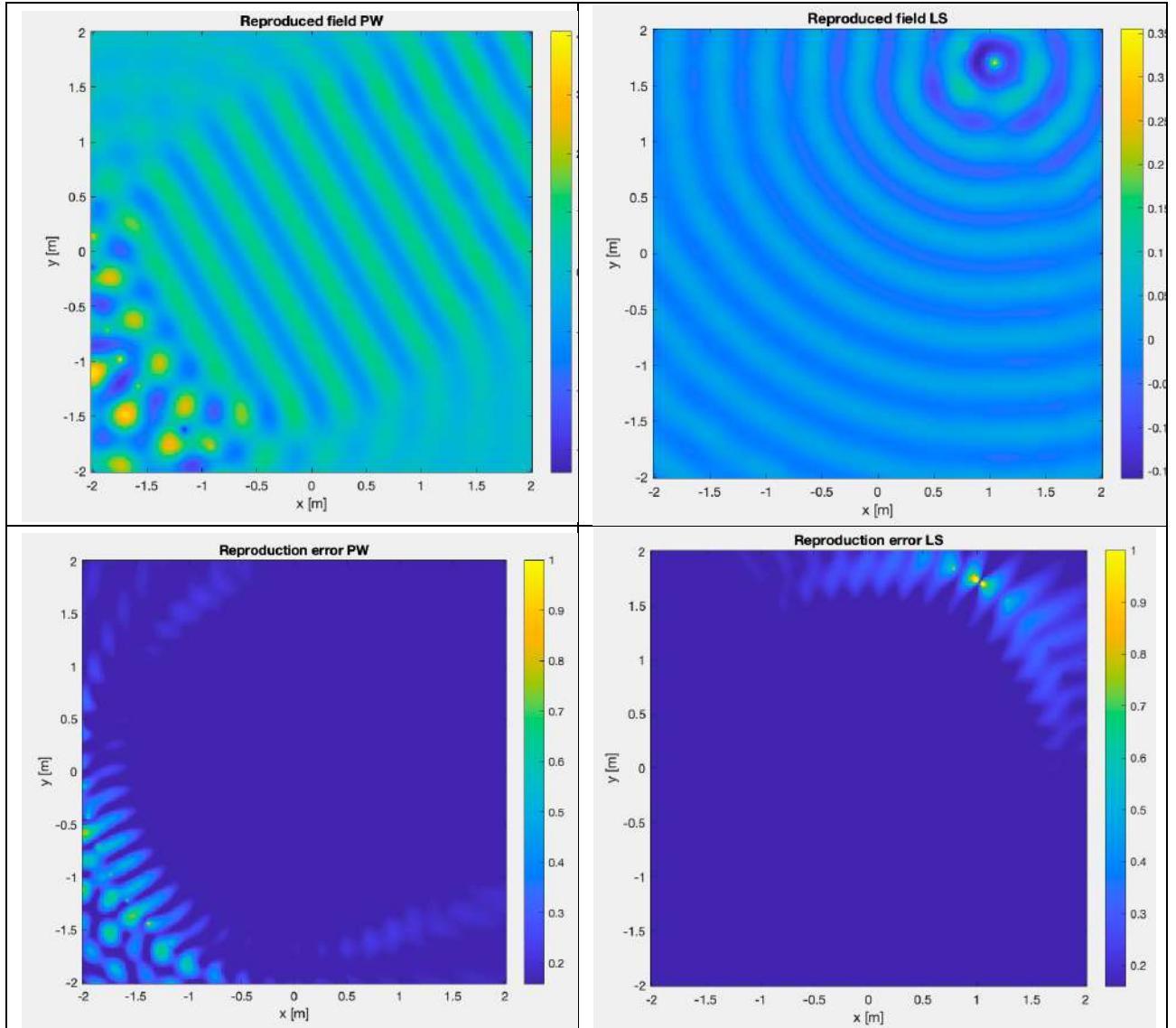
A desired plane wave field



And a desired linear source field



Are represented as it follows (with the relative error)



### 3D Geometry system

Let us switch now to a 3D scenario. Arbitrary order 3D ambisonics methods can be applied using whichever 3D microphone array geometry in the analysis phase and whichever 3D loudspeaker array geometry in the rendering phase. It is not mandatory that microphone array geometry has to be the same of the loudspeaker array ones. However, the most widely used geometry both for microphones and loudspeakers is the spherical array.

In the two following pictures, we can see some examples of them:

Here there is a picture of a semi-sphere ambisonics 3d system in Como.



Here we have the professor Zanoni inside an ambisonics system in Cremona.



For the sake of simplicity, but without loss of generality, we make the following assumptions

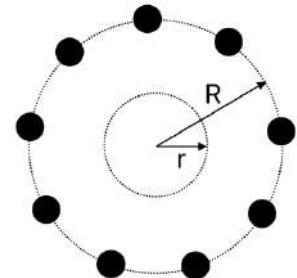
- We consider a distribution of microphones placed on the surface of a sphere during the recording phase
- We consider a distribution of loudspeakers placed on the surface of a sphere with radius  $R$  during the rendering phase

We are interested in representing the sound field in a spherical area (defined as the listening area or sweet spot) inside the sphere of loudspeakers.

Mathematically, the radial coordinate of points inside the listening area that satisfies  $r < R$ .

How can we represent such a sound field?

As an infinite summation on all the terms and on all the orders (also called degrees)



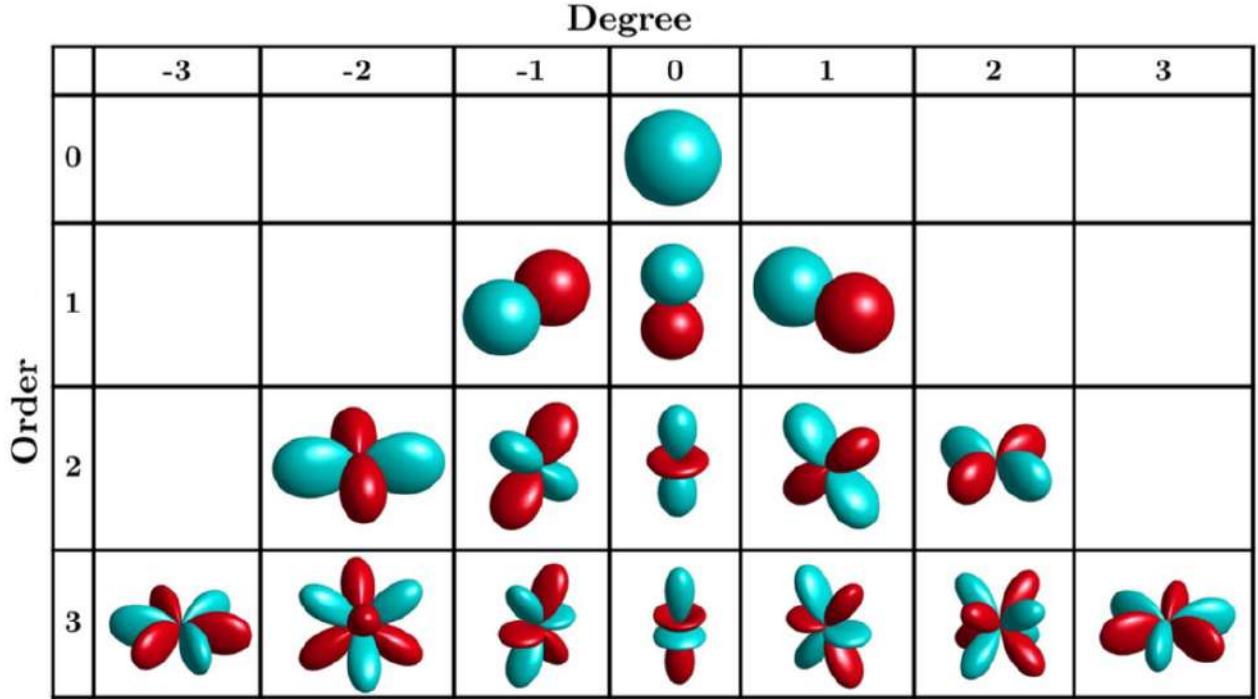
$$p(\underline{r}, \omega) = \sum_{n=0}^{\infty} \sum_{m=-n}^n \alpha_{nm}(\omega) j_n \left( \frac{\omega}{c} r \right) Y_n^m(\cos(\phi))$$

This time again, we have to truncate the presentation as we are finite, and we cannot deal with infinite sum. We truncate it to  $|n| \leq N$ :

$$p(\underline{r}, \omega) = \sum_{n=0}^N \sum_{m=-n}^n \alpha_{nm}(\omega) j_n \left( \frac{\omega}{c} r \right) Y_n^m(\cos(\phi))$$

What we have here is an expansion in terms of circular harmonics.

We remind that the spherical harmonics  $Y_n^m(\cdot)$  are obtained as the product between a spherical Bessel function of order  $n$  and a unit amplitude spherical harmonic.



We remind that the higher the order of the spherical harmonics, the more complicated becomes its shape. We also remind that given the order  $N$ , we will need  $(N + 1)^2$  coefficients to represent it.

And now, guess what? We have to find the coefficients  $\alpha_{nm}(\omega)$  such that a target desired sound field  $p_{target}(\underline{r}, \omega)$  can be approximated as

$$p_{target}(\underline{r}, \omega) = \sum_{n=0}^N \sum_{m=-n}^n \alpha_{nm}(\omega) j_n \left( \frac{\omega}{c} r \right) Y_n^m(\cos(\phi))$$

These coefficients can be found in different ways according to the scenario we are considering:

- In a model-based scenario,  $\alpha_{nm}(\omega)$  depends on the model for the desired sound field
- In a data-based scenario,  $\alpha_{nm}(\omega)$  are inferred from the recorded microphone signals

Let us see these two scenarios:

### ***Model based-scenario***

The procedure is exactly the one we did before. We will not see the details. What we get at the end is that

- For plane waves coming from direction  $(\theta_z, \phi_z)$  the coefficients will be

$$\alpha_{nm}(\omega) = 4\pi(-j)^m Y_n^{-m}(\theta_z, \phi_z)$$

- For a point source located at  $\underline{z} = (z, \theta_z, \phi_z)$  the coefficients will be

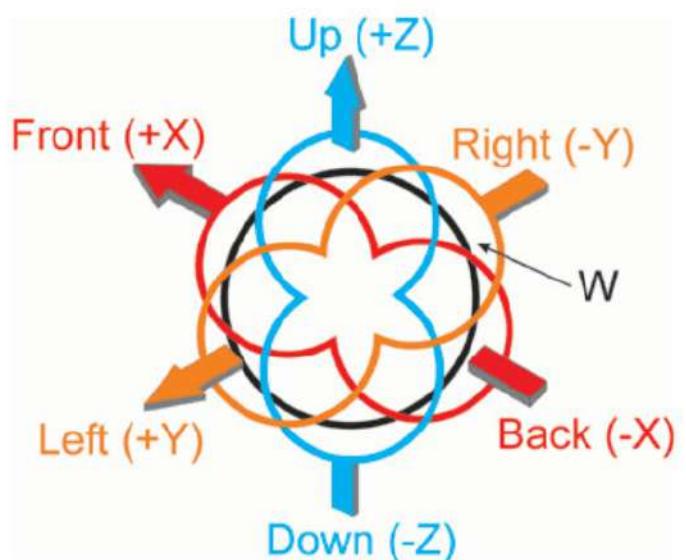
$$\alpha_{nm}(\omega) = -j \frac{\omega}{c} h_n^{(2)} \left( \frac{\omega}{c} z \right) Y_n^m(\theta_z, \phi_z)$$

### *Data-based scenario: first-order ambisonics*

What happens in reality is that we use a data-based scenario. Let us see the first-order ambisonics system. A special microphone called sound field microphone should be required during the acquisition phase. The most common format is the B format, a microphone that acquires:

- Pressure signal  $W$  (out of omnidirectional microphone)
- Directional signal  $X$  in the direction of  $x$  axis
- Directional signal  $Y$  in the direction of  $y$  axis
- Directional signal  $Z$  in the direction of  $z$  axis

In real application, we use linear arrays microphone, and we apply a transformation to the signals to obtain the b-format result. We report here an example where we have cardioid capsules whose signals are transformed in monopole signals and dipoles signals



### *Data-based scenario: HOA*

An appropriate higher-order representation of the captured sound field can be recreated with an approach similar to the one used in first-order ambisonics. Since the order is higher, a higher number of coefficients is required. The desired sound field is expressed by the expansion coefficients of the captured one. In this case, we need high order microphone directivity (first-order cardioid is not enough). We use, in fact microphone arrays and spherical microphone arrays (the most widespread since they provide equal properties for all angles of sound incidence). This kind of microphones are implemented mounting sensors on a spherical surface that allow to capture each spherical harmonic wave independently.



### **Data-based scenario in general**

As we saw, the sound field scenario at point  $\underline{r}$  in terms of spherical harmonics of order  $n$  and degree  $m$  can be expressed by

$$p(\underline{r}, \omega) = \sum_{n=0}^{\infty} \sum_{m=-n}^n \alpha_{nm}(\omega) j_n\left(\frac{\omega}{c} r\right) Y_n^m(\theta, \phi)$$

Where coefficients  $\alpha_{nm}(\omega)$  are independent of location. If we are able to retrieve those coefficients by the signals recorded, we can reconstruct the sound field accurately. Ideally, those coefficients by means of the inverse formula noticing that is nothing but a spherical integration

$$\alpha_{nm}(\omega) = \frac{1}{j_n\left(\frac{\omega}{c} r\right)} \int_0^{2\pi} \int_0^\pi p(\underline{r}, \omega) Y_n^{-m}(\theta, \phi) d\theta d\phi$$

Expression that is valid only under the following conditions:

- $\underline{r}$  is on the surface on a sphere of radius  $r = R$
- The first order spherical Bessel function  $j_n\left(\frac{\omega}{c} r\right) \neq 0$ . This means that there are no coefficients for the points in which the Bessel function cross the zero axis.

In order to approximate the integration by finite summation, we sample the sphere. This is done by using  $\tilde{Q}$  omnidirectional microphones on a sphere of radius  $R$ . The sound field that we will obtain will be in function of the discretised spherical coordinates  $(R, \theta_q, \phi_q)$  with  $q = 1, \dots, \tilde{Q}$ , so  $p(R, \theta_q, \phi_q, \omega)$ . The coefficients of such a sound field will be, in turn, in function of those discretised spherical coordinates

$$\hat{\alpha}_{nm}(\omega) = \frac{1}{j_n\left(\frac{\omega}{c} r\right)} \sum_{q=1}^{\tilde{Q}} p(R, \theta_q, \phi_q, \omega) Y_n^{-m}(\theta, \phi) w_q$$

Where  $w_q = \frac{2\pi}{\tilde{Q}}$  are suitable weights needed for the discretisation.

But hey hey hey, gattona! With an equiangular sampling on a sphere, we end up with a finer sampling around the poles and a less dense sampling in the equatorial region due the spherical structure. This must be taken into account when we design the system.

In light of all these considerations, how many microphones do we need to implement an ambisonics system (how much has to be equal  $\tilde{Q}$  to?)

Naïvely, the system has to capture all the harmonics components. We know that a sound field bandlimited to  $N$  has  $(N + 1)^2$  harmonic components. This means that we should need a number of microphones at least equal to  $(N + 1)^2$

$$\tilde{Q} \geq (N + 1)^2$$

As we said, this is the naïve solution in which we do not consider the fact that we have a denser packing near poles. The correct expression is given by

$$\tilde{Q} \geq (2N + 1)^2$$

But, what about the frequency band? We know that spherical Bessel's functions cross zero axis in several points. We need to modify the frequency band in order to avoid this behaviour. But what are we saying? We are talking about hi-fi systems, and we throw away a set of frequency with such a light heart. Obviously we need to find another solution. The spherical Bessel function takes two variables and a constant as arguments: they are the frequency, the ray  $R$  and the speed of sound. We cannot change the constant, we cannot throw away frequencies, we cannot do nothing but choose a ray  $R$  in a way that

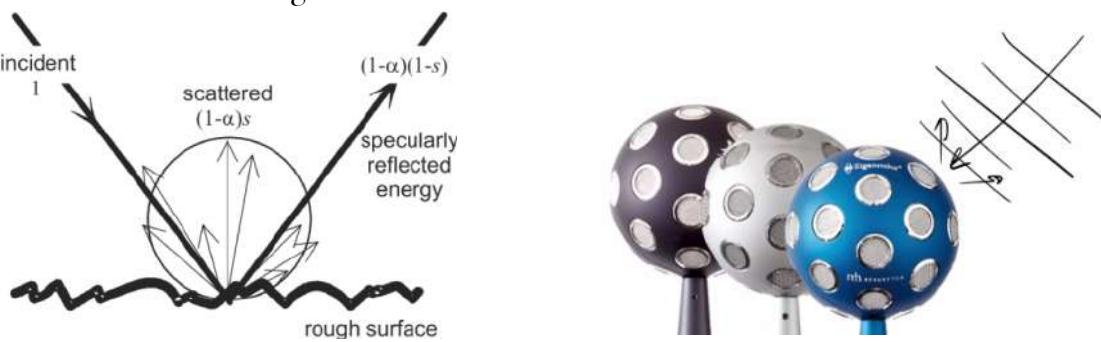
$$j_n\left(\frac{\omega}{c}R\right) \neq 0 \quad \text{for } n = 0, \dots, N \text{ and } \omega \in [\omega_l, \omega_u]$$

Since we cannot design a new microphone for every application, to overcome this limitation, microphones are placed on the surface of a rigid sphere with a standardised ray. The fact that we have rigid sphere generates the scattering phenomena that introduces into the equations some terms that prevents the Bessel function from going to zero. Fortunately, scattering is a model that can be modelled analytically; therefore, we are able to effectively add some terms to our equation even though this does not help us very much: the model becomes so complicate (as scattering depends on the geometry, on the material of the sphere and so on) that we need to do a number of approximation that, at the end, makes the mode not so accurate. One of the most important microphones that is based on this principle is the Eigenmike by MH Acoustics, reported in the picture on the right.



#### Reminder: Scattering

Considering an impinging wave on a surface, Scattering occurs when a sound wave strikes a structure with a different acoustic impedance to the surrounding tissue, and which is smaller than the wavelength of the incident sound wave. This generates several reflections with different directions.



Of course, the ambisonics coefficient can be made explicit in the rendering phase. The weights at the  $q$ th loudspeaker on the position  $\underline{v}$  will be expressed by

$$d_q(\underline{v}, \omega) = \sum_{n=0}^N \sum_{m=-n}^n -\frac{j\alpha_{nm}}{\frac{\omega}{c} h_n^2 \left( \left( \frac{\omega}{c} \right) R \right)} Y_n^m(\theta_q, \phi_q) \Delta$$

Some restrictions again: the spherical distance between adjacent loudspeakers has to be equal to

$$\Delta = 2\pi Rh \quad \text{with } h = R - \cos\left(\frac{\Delta\phi}{2}\right)R$$

in which  $\Delta\phi$  is the minimum angle between adjacent speakers on the sphere.

## Recap on the differences between wavefield synthesis and ambisonics

Let us consider a 2D scenario

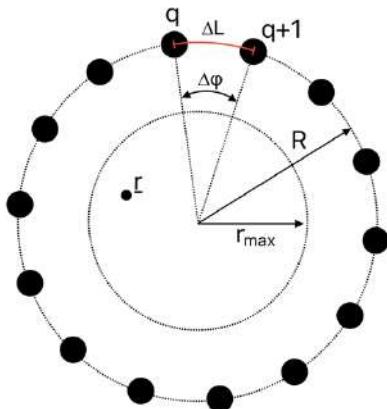
*2D Scenario:*

A circular distribution of secondary sources with radius  $R$  spatially sampled by a series of loudspeaker. We call  $\Delta l$  the arc between two consecutive loudspeakers. We call  $\Delta\phi$  the angular displacement between two loudspeakers that are close by.

$$\Delta l = \Delta\phi \cdot R$$

With  $\Delta\phi = 2\pi Q$

The listening area of ray  $r_{max}$  is inside the circle of radius  $R$ . There, we have a listener positioned in  $\underline{r}$ .



### Wavefield Synthesis

In wave field synthesis,  $R = r_{MAX}$ : the reproduction region is larger.

If we increase  $R$ , we increase  $Q$  (where  $Q$  is the number of loudspeakers). In fact, to obtain the same sound field rendering accuracy (respecting the aliasing requirement  $\Delta L \leq \frac{\lambda}{2}$ ) in a larger area, we have to increase the number of loudspeakers.

### Ambisonics

In the ambisonics technique  $r_{MAX} < R$ . Here we can move the loudspeakers as far as we like but we will need the same number of loudspeakers. Even if we increase  $R$ ,  $Q$  remains the same. The only constraint that we have to take into account is that

$$\Delta\phi = -\frac{2\pi}{2M + 1} \quad \text{with } M = \left\lceil \frac{er\omega}{2c} \right\rceil$$

## References:

- [1] E. G. Williams. Fourier Acoustics. Academic Press, London, UK, 1999.
- [2] R. A. Kennedy, P. Sadeghi, T. D. Abhayapala, and H. M. Jones. Intrinsic limits of dimensionality and richness in random multipath fields. *IEEE Transactions on Signal Processing*, 55(6):2542–2556, June 2007.
- [3] D. Colton and R. Kress. Inverse Acoustic and Electromagnetic Scattering Theory. Springer-Verlag, Berlin Heidelberg, DE, 1992.
- [4] N. Xiang and C. Landschoot. Bayesian inference for acoustic direction of arrival analysis using spherical harmonics. *Entropy*, 21, 2019.
- [5] T. D. Abhayapala and D. B. Ward. Theory and design of high order sound field microphones using spherical microphone array. In Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP), 2002.
- [6] S. Spors, H. Wierstorf, A. Raake, F. Melchior, M. Frank, and F. Zotter. Spatial sound with loudspeakers and its perception: A review of the current state. *Proceedings of the IEEE*, 101(9):1920–1938, 2013.
- [7] Y. J. Wu and T. D. Abhayapala. Theory and design of soundfield reproduction using continuous loudspeaker concept. *IEEE Transactions on Audio, Speech and Language Processing*, 17(1):107–116, 2009.
- [8] M. Abramowitz and I. A. Stegun, editors. Handbook of Mathematical Functions. National Bureau of Standards, Washington DC, USA, tenth edition, 1972.
- [9] F. W. J. Olver, editor. NIST Handbook of Mathematical Functions. National Institute of Standards and Technology, New York, NY, USA, 2010.
- [10] J. Meyer and G. Elko. A highly scalable spherical microphone array based on an orthonormal decomposition of the soundfield. In Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP), 2002.
- [11] B. Rafaely. Analysis and design of spherical microphone arrays. *IEEE Transactions on Speech and Audio Processing*, 13(1):135–143, Jan. 2005.

**Master:** ENI for underground application of sound engineering We are getting pay €1500 to attend it.

## Reverberation Algorithms

In the following lectures we will see:

- How to synthetise reverberation
- How to estimate reverberation parameters from real environment.

The idea of reverberation in audio signal goes back to the 60s. The concept of reverberation can be considered closed to echo but with a strong difference: reverberations are so many and so dense that you are not able to pick them separately.

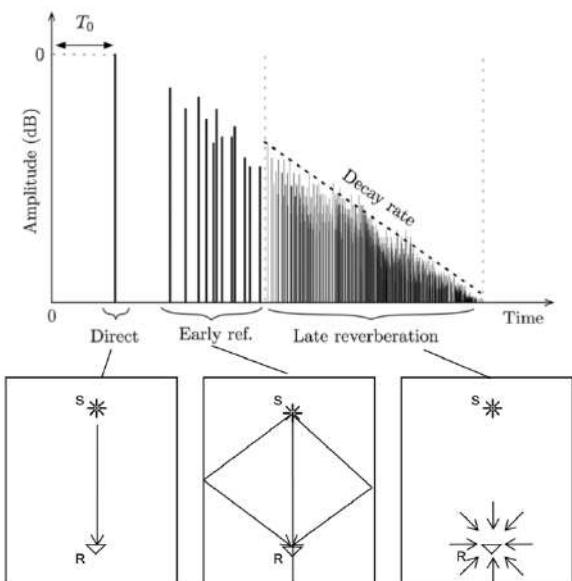
Why are we interested in introducing reverberation in sound? On one hand, since we are so used to it, listening to a completely dry sound sounds so unnatural. On the other hand, to create different acoustical environments.

### History:

A story of reverberation before DSP theory. The need for artificial reverberation first arose in the context of music broadcasting and recording. As early as the 1920s, the strategy was to reproduce the sound in an echo chamber and re-record the sound inside it. The problem was that they had to actually build this echo chambers: huge, affected by environmental conditions. Then, plate reverb comes in to replace these echo chambers. From other interesting objects, we can find the echo fix, a tape reverb, if we have 24 hundred euros left apart...



### Impulse response:



Three different stages can be identified in the impulse response:

- Direct
- Early reflections
- Late reverberation: so many decaying peaks very dense. After so many bounces, we receive the final sound. They give information about the aesthetic of an environment.

How we can imagine, modelling the first or the final part of reverberation requires different tools.

Some consideration. A signal traveling along a direct path between the source and listener arrives after a delay of  $T_0$  and is followed by early reflections from nearby objects and surfaces. While the direct path reveals the

direction of the source, the early reflections convey a sense of the geometry and materials of the space.

Reflections propagating in the environment themselves interact with features of the space to generate additional reflections, and therefore produce an increasing echo density. Over time, echoes form a dense late reverberation tail, characterized by Gaussian statistics and an evolving power spectrum. As we said, this late reverberation stage reveals more qualitative features of the environment's acoustics (size of the space, absorbing power of the materials, pleasantness of the response).

The process of reverberation is approximately linear and time invariant (LTI), therefore it is of convolutive nature. The focus is on reproducing the psychoacoustic impact of various reverberation impulse response features. Reverberation algorithms generally fall into one of three categories:

1. **Delay networks:** signal is delayed, filtered and fed back along a number of paths according to parametrized reverberation characteristics;
2. **Convolutional:** signal is convolved with a recorded/estimated Room Impulse Response (RIR);
3. **Computational acoustic:** signal drives a simulation of acoustic energy propagation in the modelled geometry.

What to use depends on the computational setting and the application. Delay-network and convolution techniques have been competing in popularity in the music technology field, and they are often employed to produce a desired perceptual or artistic effect. Immersive audio needs more...

## Descriptors of reverberation

Before digging in the different family of reverberations system, we spend some words on how to describe them.

- ***Energy decay curve***

It is a curve that for each time instant represents the remaining energy of the impulse response. It is a very good parameter except that it does not work! The noise taken into account makes the curves weird. For example, the first curve decays with a certain slope but a certain point it changes. This is due to the noise floor that, integrate, changes the slope of the function. Practically, this noise floor increases the decay time of the reverb.

$$h_{EDC}(t) = \int_t^\infty h^2(\tau) d\tau$$

- ***Measuring the T<sub>60</sub>***

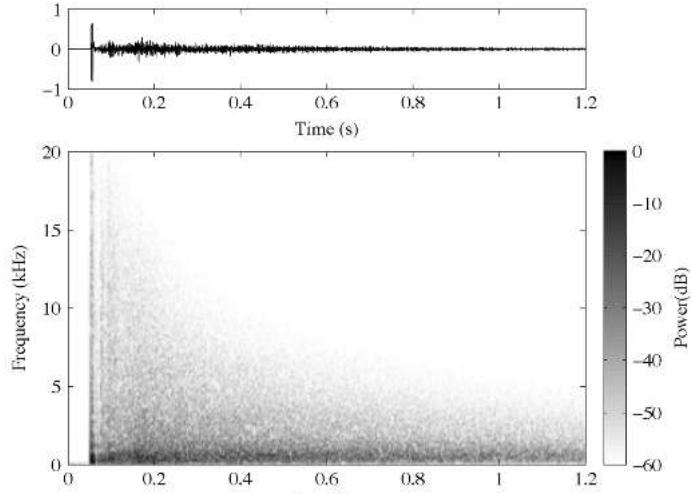
$$T_{60} = \{t: [h_{EDC}(t)]_{dB} = [h_{EDC}(0)]_{dB} - 60\}$$

Noise conditions usually are not good enough as to enable the measurement of the T<sub>60</sub>, therefore sometimes we have to resort to alternate definitions...

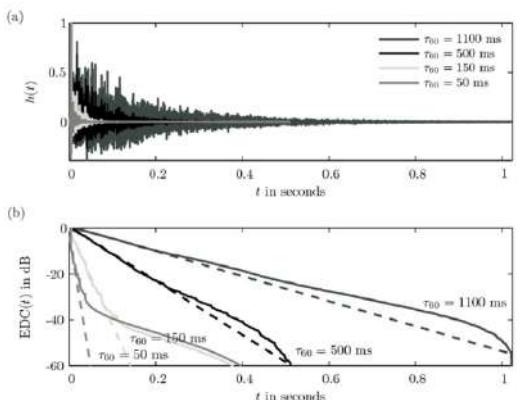
$$T_{40} = \{t: [h_{EDC}(t)]_{dB} = [h_{EDC}(0)]_{dB} - 40\}$$

$$T_{20} = \{t: [h_{EDC}(t)]_{dB} = [h_{EDC}(0)]_{dB} - 20\}$$

Knowing that  $T_{60} \approx T_{40} \approx T_{20}$



***Impulse response in a large Church***

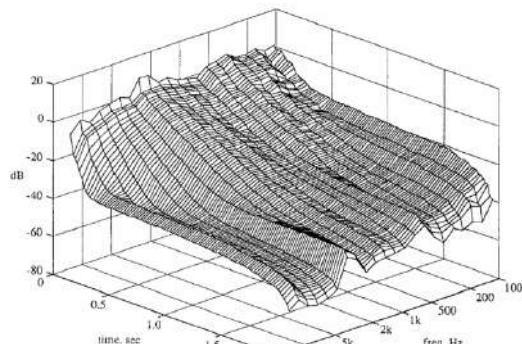


- **Energy Decay Relief (EDR)**

This is nothing but the energy decay curve measured frequency by frequency.

$$H_{EDR}(n, k) \triangleq \sum_{m=n}^M |H(m, k)|^2$$

It is sometimes useful for other applications (for example for the input response of a violin body) even though it is a bit of overpassed technique as it does not take into account the non-linearities.



Energy decay relief for occupied Boston Symphony Hall

- **Centre time**

Centroid of the squared impulse response

$$T_s = \frac{\int_0^\infty \tau \cdot h^2(\tau) d\tau}{\int_0^\infty h^2(\tau) d\tau} [s]$$

- **Clarity index**

It used to quantify the intelligibility of a speech in a large environment

$$C_{t_e} = 10 \log_{10} \left( \frac{\int_0^{t_e} h^2(\tau) d\tau}{\int_{t_e}^\infty h^2(\tau) d\tau} \right) [dB]$$

If the reverberation is long, we have most of the energy after  $t_e$ . Therefore  $C_{t_e}$  is getting smaller. On the other hand, it is getting higher.

- **Definition index**

It is kind of the opposite of the clarity index and not in dB.

$$D_{50} = \frac{\int_0^{50ms} h^2(\tau) d\tau}{\int_0^\infty h^2(\tau) d\tau} [\%]$$

- **Other parameters**

In the Room's Transfer Function (RTF) we can single out resonant modes, and we find that the spacing between two nearby resonant modes can be roughly estimated as

$$\Delta f_{\max} \approx \frac{4}{T_{60}} \quad \text{for } f > f_g \approx 2000 \sqrt{\frac{T_{60}}{V}} \quad \text{where } V \text{ is the Volume}$$

In the RIR, the number of echoes in the impulse response before time  $t$  can be roughly estimated as  $N_t = \frac{4\pi(ct)^3}{3V}$ , while its derivative (density of echoes over time) is given by

$$\frac{dN_t}{dt} = \frac{4\pi c^3}{V} t^2$$

He would lie if he told us that this parameter will become clear later.

## The Arvedi Auditorium in Cremona

It is an auditorium though for solo concertos. It was designed by a Japanese architect with a weird shape. It is an example of how art and experience meet maths and engineering. The irregularity of the room is done inspired by the experience of the architect.

When we want to measure the response of an environment, the explosion is not the best option as we cannot control the energy associated to each frequency. That is why they used different methods.



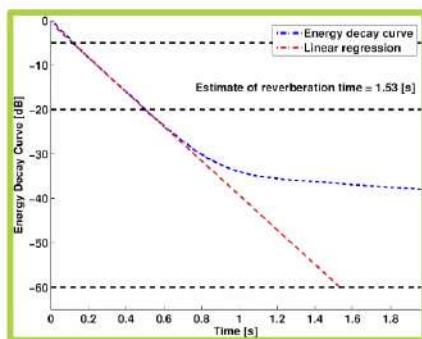
We report some results, just out of curiosity.

### Reverberation Time (T60)

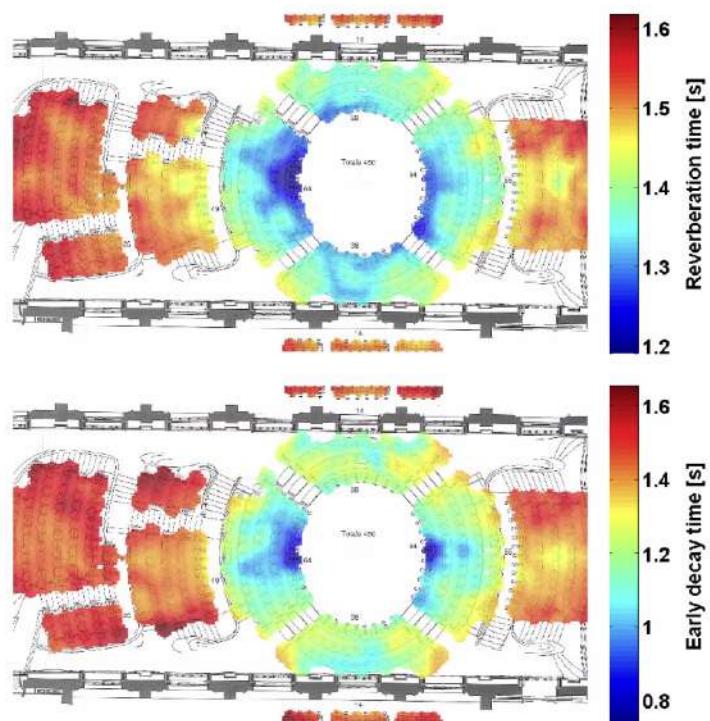
Estimated from T15  
(optimal value of T60 = from 1.3 to 2s)\*

### Early Decay Time (EDT)

Slope of the decay of the EDC btw 0 dB and 10dB

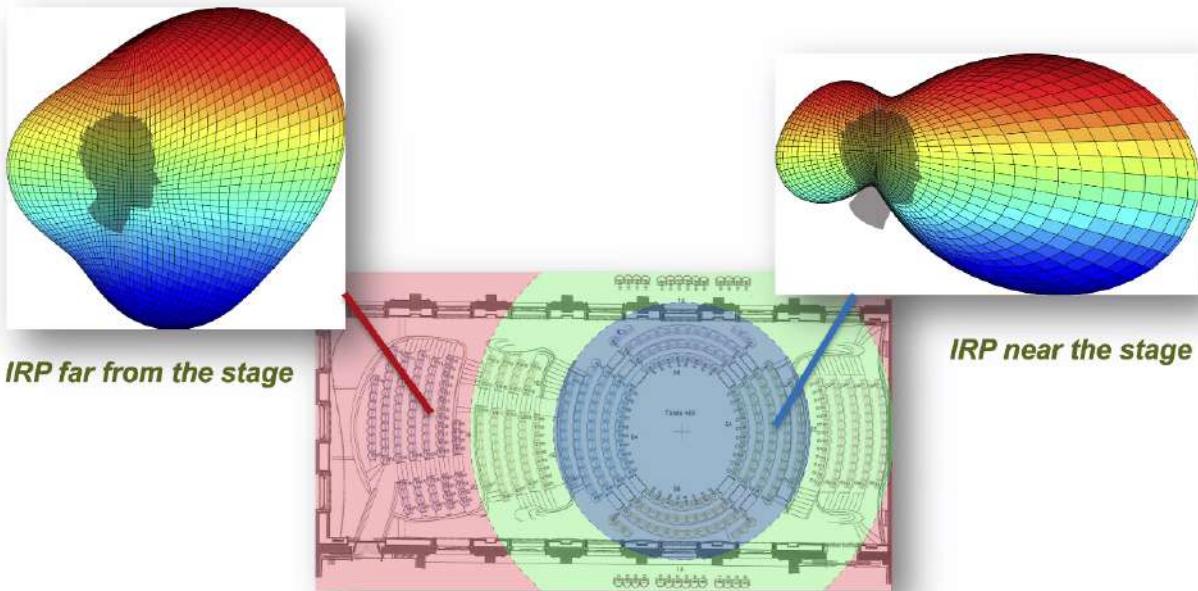


\* Range for concert halls and classical music



Also some results on intelligibility and clarity are reported.

- Measurements based on **beamforming**



If we are closer to the orchestra, we can understand what a single musician is doing. To have the better overall experience is better to be far away.

Some interesting applications can be also found in forensic analysis.

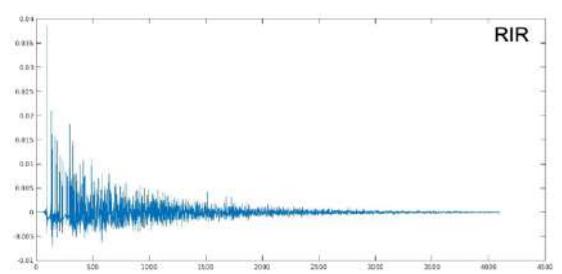
### The reverb problem

Now that we know how to characterise a reverb, let us go back to reverberation problem: how to add reverb to a dry sound.

Let us do some maths. Let us suppose that we are creating a reverb of a  $T_{60} = 1$  s and a sampling frequency of  $f_s = 50$  kHz. Each filter requires 50000 multiplications ad additions per sample. This means that we briefly reach 5 billion operations. Having three sources and 2 listening points (as the ears) we have to do at least 30 billion operations. The number is huge and extremely expansive. In the frequency domain we have also to perform the fft and go back, introducing delay.

But the question is, do we really need to apply to a sound a reverberation that models each and every each of the environment? No, at the end of the day we do not evaluate the reverberation with maths or whatever but with our ears. So, we can focus not on the tiny details but on the perceptual aspect of the reverberation. Echo density typically grows as  $t^2$ , mode density typically grows as  $f^2$ . After some time, the echo density is so large that a stochastic process results. This means that there is no need to simulate too many echoes per sample. Above some frequency, the mode density is so large that a random frequency response results, having no need to implement more resonances than the ear can perceive. Based on limits of perception, the RIR impulse response of a reverberant room can be divided into two segments

- Early reflections = relatively sparse first echoes
- Late reverberation—so densely populated with echoes that it is best to characterize the response statistically.



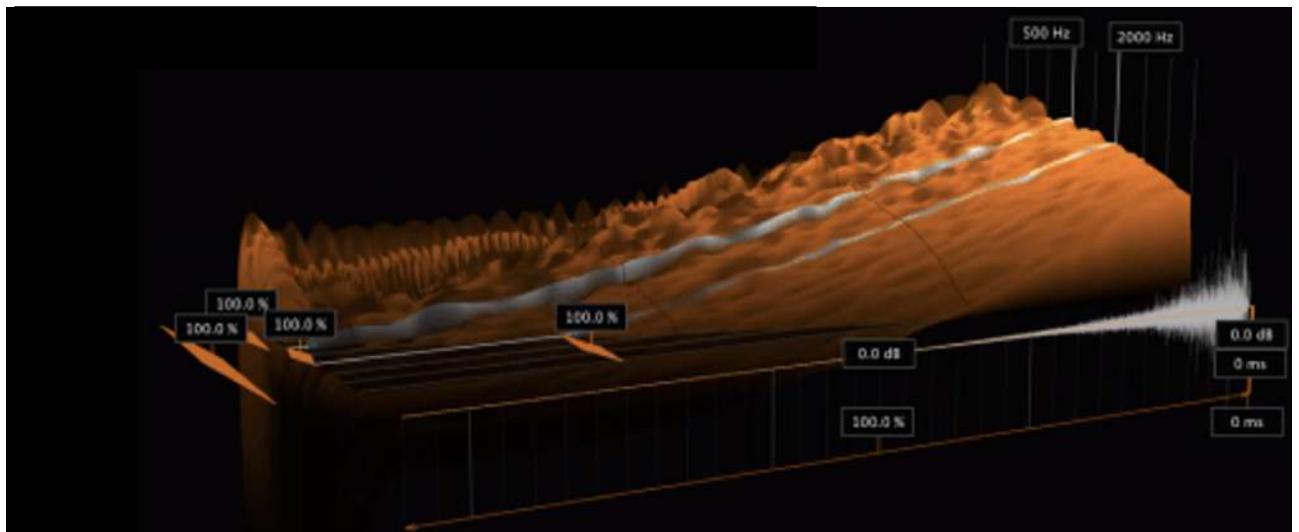
Similarly, the frequency response of a reverberant room can be divided into two segments.

- Low-frequency sparse distribution of resonant modes
- Modes packed so densely that they merge to form a random frequency response with regular statistical properties



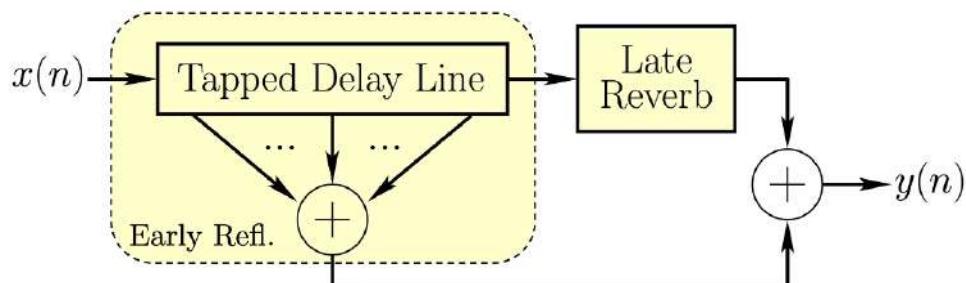
We want to model a reverberation where early reverberations are and the same goes for the frequency response. Therefore, we focus on just few parameters that are perceptually relevant:

- $T_{60}(f)$ : desired reverberation time at each frequency (at least 3 frequency bands)
- $G^2(f)$ : signal power gain at each frequency
- $C(f)$ : clarity index
- $\rho(f)$ : inter-aural correlation coefficient at left and right ears.



### Reverberation algorithms

The model we choose is, therefore, the one in which the reverb is generated as a sum of early reflections and a later reverb.

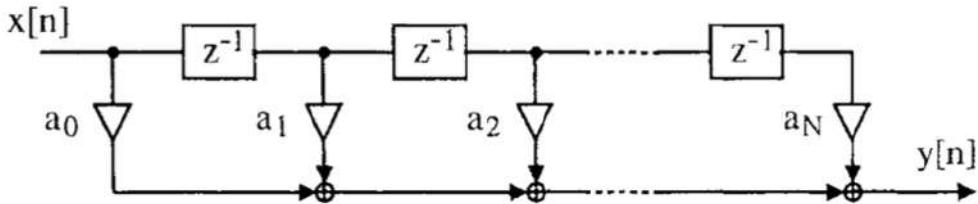


The “early reflections” portion of the impulse response is defined as everything up to the point at which a statistical description of the late reverb takes hold (often takes to be the first 100 ms). It is typically implemented using tapped delay lines (TDL). We highlight that the early reflections should be spatialised as they influence strongly the spatial impression.

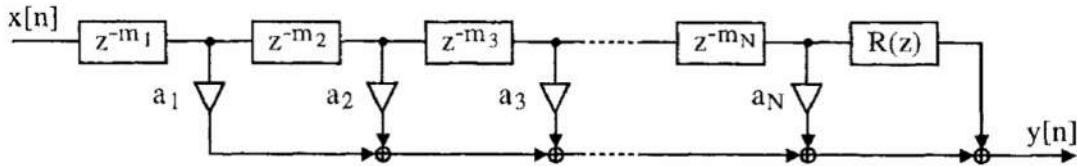
## Early reflections + Late Reverb

### *Early reflections*

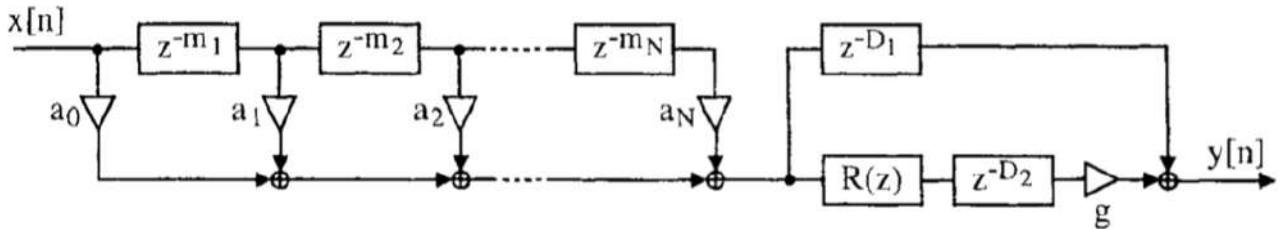
The first model we propose to implement a reverb is the following. We start from the canonical direct form FIR filter with single sample delays (as we can imagine, due to its structure, this method is inefficient for long RIR).



As echoes are sparse Schroeder proposed a Tapped Delay Line (TDL) to generate a specific pattern of early echoes combined with a late diffuse reverberation block.



Moorer proposed a different structure, where the late reverb is driven by the output of an early echo TDL, in order to increase the echo density of the late reverberation.



Delays  $D_1$  and  $D_2$  can be adjusted so that the first pulse output from the late reverberator corresponds with the last pulse output from the FIR section. The gain  $g$  serves to balance the amount of late reverberation with respect to the early echoes.

As we make lots of useless multiplication by zero, we can add a sort of degree of control to our reverberation system. We can also add a differentiation of the right and left channel. It can become as complex as we want.

### *Late Reverberations*

Let us see now how to model the late reverberations. What we need are a smooth (but not too smooth) decay and a smooth (but not too regular) frequency response. Exponential decay is not a problem. The hard part is making it smooth. We have to avoid “flutter,” “beating,” or unnatural irregularities. A smooth decay needs high echo density. Despite that, we have still to include some short-term energy fluctuation for naturalness. Furthermore, we have to provide a smooth frequency response with no large “gaps” or “hills”. This is generally provided when the mode density is sufficiently large. Modes should be spread out uniformly, but modes may not be too regularly spaced, since audible periodicity in the time-domain can result.

Moorer's propose an ideal late reverb: exponentially decaying white noise. It has good smoothness in both time and frequency domains and high frequencies need to decay faster than low frequencies.

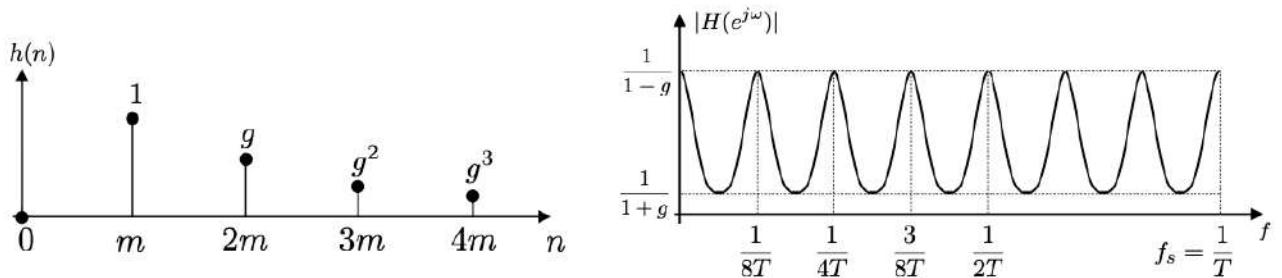
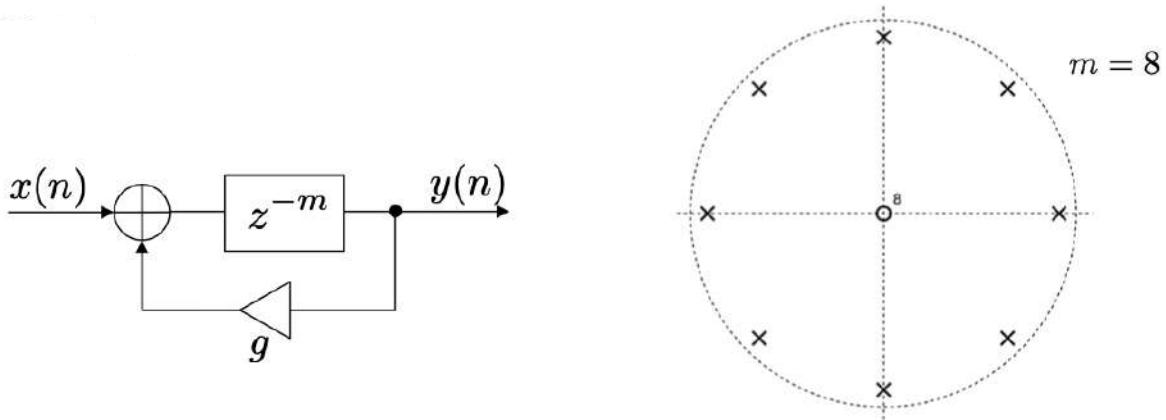
Along that, Schroeder's rule of thumb for echo density is the follower

- late reverb: 1000 echoes/s or more
- impulsive sounds: 10 000 echoes/s or more (for a smooth response)

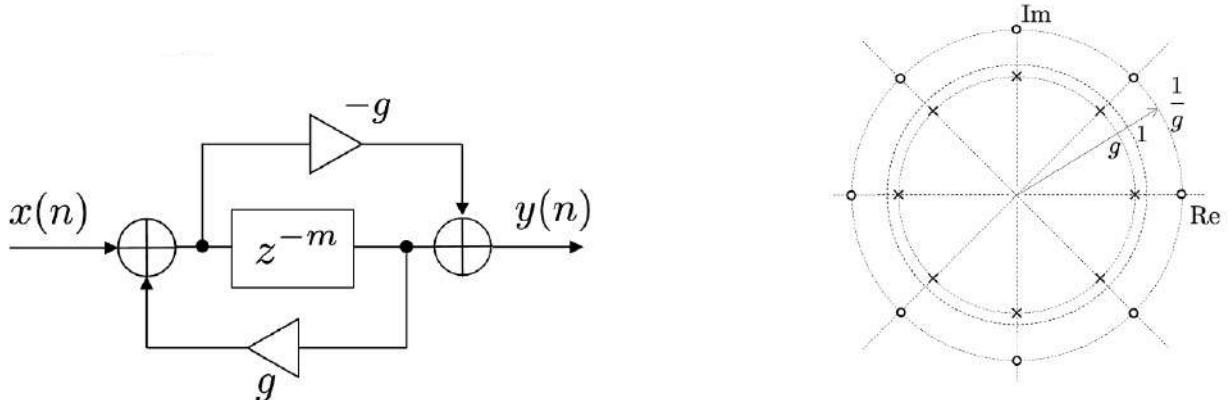
So, in order to implement also the late reverberation, we use the tapped delay lines as building blocks for COMB filter.

$$y(n) = gy(n-m) + x(n-m)$$

$$(1 - gz^{-m})Y(z) = z^{-m}X(z) \rightarrow H(z) = \frac{Y(z)}{X(z)} = \frac{z^{-m}}{1 + gz^{-m}}$$



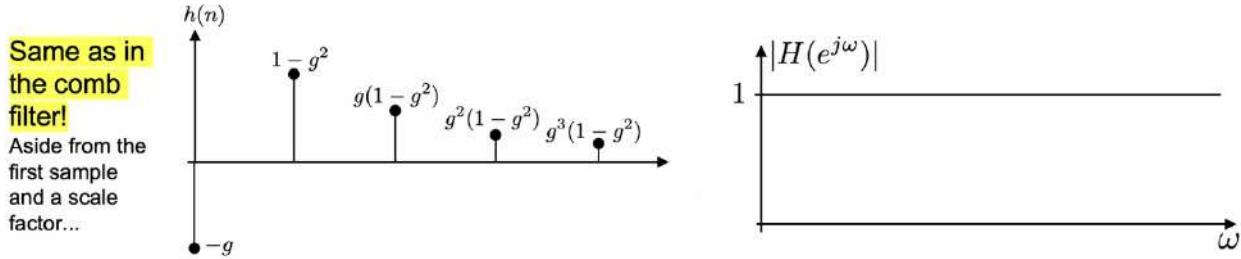
If I look at the impulse response of this structure here, I find something that is exponentially decaying. It is exactly what we want! Obviously, this happens if and only if  $g < 1$ . The roots will be radially distributed on a circle of radius  $g$ . This is the simplest response that we can have but it adds some colour to the sound as  $|H(e^{j\omega})|$  is not flat. We need something that is similar to the comb filter but that does not modify the spectrum. In principle, we can use an all-pass filter.



$$y(n) - gy(n-m) = -gx(n) + x(n-m)$$

$$(1 - gz^{-m})Y(z) = (-g + z^{-m})X(z) \rightarrow H(z) = \frac{Y(z)}{X(z)} = \frac{1 - gz^{-m}}{-g + z^{-m}}$$

It is just a comb filter with a product and a sum more. The gain  $g$  has to be smaller than one to have all the poles inside the circle.



Whatever is the frequency of the signal that I am feeding to my all-pass filter, the response is the same. It behaves still pretty nicely in the time domain. As we notice, it is almost identical to the comb filter. But, using an all-pass filter brings us to the problem of the all pass filter.

- The impulse response is extremely long (infinite). In order to have a response that is actually flat I need infinite samples.
- Our ears perform a short-time frequency analysis, while the frequency properties of the APF are defined over an infinite time integration.

In light of this, we could ask ourselves are APF (all-pass filters) really colourless?

The APF impulse response is perceived as “colourless” only when it is extremely short ( $< 10ms$ ). Long APF impulse responses sound like feedback comb-filters. Steady-state tones (e.g. sinusoids) really do see the same gain at every frequency in an APF, while a comb filter has widely varying gains. So, more or less.

Anyway, for our application, how do we tune the all-pass filter to have a desired reverberation time?

We have two parameters to control: the decay of the response and the echo density. For the first one we can work on the coefficient  $g$  of the comb cell. In order for this decay to match a given reverberation time we must have

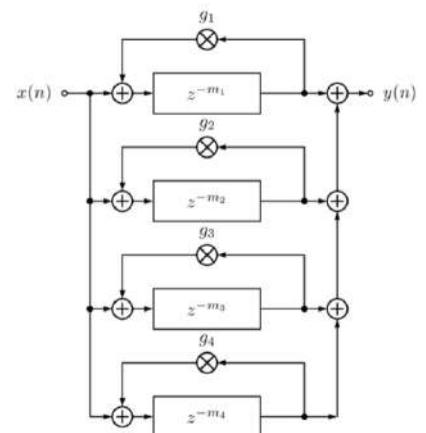
$$\frac{20 \log_{10}(g)}{mT_s} = -\frac{60}{T_{60}} \rightarrow g = 10^{-\frac{3mT_s}{T_{60}}}$$

One problem is solved but I am not happy enough as the all-pass filter does not allow us to control the echo density (the reverberant tail).

What we can do to have a more dense tail and to fix this not colourless behaviour, we can combine more filters all together either in series or in parallel. But hey! Comb filter cannot be combined in series:

(cascade) the behaviour like peaks and valleys peaks and valleys makes the multiplication (series) of the transfer function totally attenuated. We need to combine them in parallel obtaining a transfer function like the following:

$$H(z) = \sum_{i=1}^N \frac{z^{-m_i}}{1 - g_i z^{-m_i}}$$



Poles are given by solutions to the following equation

$$\prod_{i=1}^N (g_i - z^{-m_i}) = 0$$

Pole moduli are all the same

$$\gamma_i = \sqrt[m_i]{g_i} = 10^{-\frac{3T_s}{T_{60}}}$$

Assuming all the gains  $g_i$  are chosen to obtain the same reverberation time  $T_{60}$ , the pole moduli will be the same for all comb filters, therefore all the resonant modes of the parallel comb structure will decay at the same rate. If the pole moduli were not all the same, the poles with the largest moduli would resonate the longest, and these poles would determine the tonal characteristic of the late decay. In order to avoid tonal coloration in the late decay, it is therefore important to guarantee uniformity of pole modulus. When the delay lengths of the comb filters are set in such a way to share no common factors (incommensurate lengths), the pole frequencies will all be different from each other (except at frequency 0). In this case the echoes from two comb filters  $i$  and  $k$  will not overlap until sample number  $m_i m_k$ .

### Criteria for the realism of a reverberation algorithm

There are two practical criteria:

1. Firstly, we have to consider the modal density of parallel combs (number of modes per Hz)

$$D_m = \sum_{i=0}^{N-1} \tau_i = N_\tau$$

where  $\tau_i$  is the length of delay  $i$  in seconds, and  $\tau$  is the average delay length. Modal density of the parallel combs is constant for all frequencies, unlike real rooms where the modal density flattens at a value  $D_f$ . We can therefore design our parallel comb in such a way to reach that mode density. This leads to the following relation between the total length of the delays and the maximum reverberation time we wish to simulate:

$$\sum_i \tau_i = D_m > D_f \approx \frac{T_{\max}}{4}$$

Where  $T_{\max}$  is the maximum reverberation time desired.

In practice, however, we set

$$\sum_i \tau_i > T_{\max}$$

2. The echo density of the parallel combs is the sum of the echo densities of the individual combs. Each comb filter  $i$  outputs one echo per time  $\tau_i$  therefore the combined echo density, expressed as the number of echoes per second, is

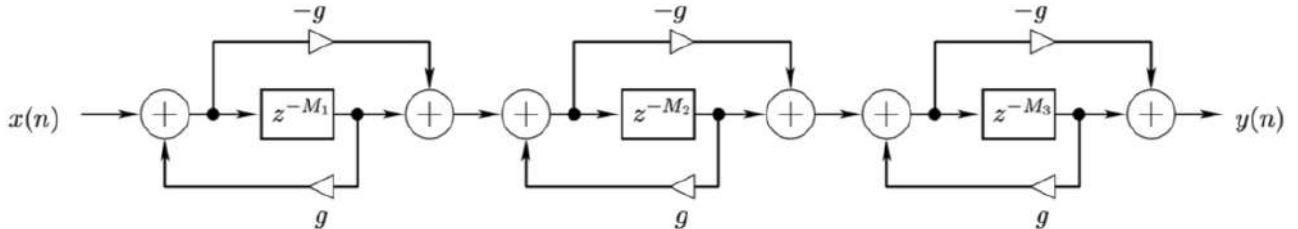
$$D_e = \sum_{i=0}^{N-1} \frac{1}{\tau_i} \approx \frac{N}{\tau}$$

which is valid when the delays are similar. Echo density is therefore constant as a function of time, unlike real rooms. According to Schröder, 1000 echoes/s are enough to sound undistinguishable from diffuse reverberation.

By putting together these two rule of thumbs, we get that the number of filters in parallel that we need to have is equal to

$$N \approx \sqrt{\mathbf{D}_m \mathbf{D}_e}$$

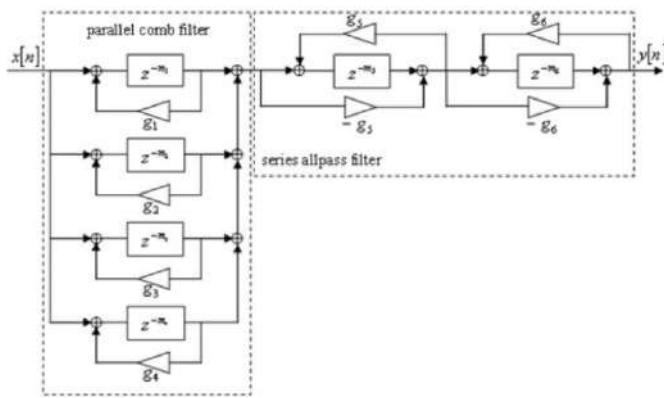
And how can we combine all pass filters?



Unlike comb filters, all-pass filters must be cascaded. The cascade of multiple APFs is an APF. Multiplying frequency responses corresponds to adding phase responses. A scheme like this one is called impulse diffuser as for every sample that we have as input we generated a decaying curve. (In this scheme, generally,  $g = 0.7$ ).

### Schröder's reverberator

At the end, we put together the two sections obtaining the so-called Schröder's reverberator



- Delays of the comb and all-pass filters are chosen so that the ratio of the largest and smallest delay is 1.5 (typically 30 and 45 ms)
- The gains  $g_i$  of the comb filters are chosen to provide a desired reverberation time  $T_{60}$  according to

$$g_i = 10^{-\frac{3m_i T_s}{T_{60}}}$$

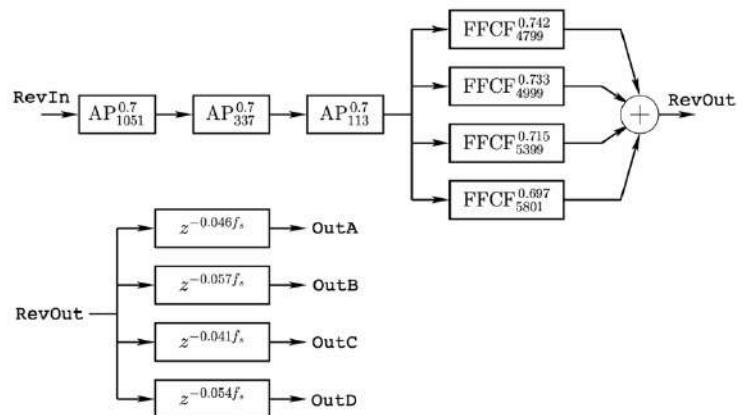
- All-pass filters delays are set to 5 and 1.7 ms

### JCRev

A slight upgrade of the Schroeder's reverberator is the JCRev. It was developed by John Chowing and others at CCRMA (<https://ccrma.stanford.edu>) (Uni-Stanford). It is composed by

- 3 Schroeder all-pass sections
- $$AP_N^g \triangleq \frac{g + z^{-N}}{1 + gz^{-N}}$$
- 4 feed-forward comb-filters
- $$FFCF_N^g \triangleq g + z^{-N}$$
- Schroeder suggests a progression of delays close to
- $$M_i T \approx \frac{100 \text{ ms}}{3^i}, \quad i = 0, 1, 2, 3, 4$$

Where a feed-forward filter is a filter containing only zeros and not poles, it is the inverse of the comb filter in terms of frequency response. Anyway, what is interesting is the output: as we



notice, we have multiple outputs which are slightly different delayed version of the same output. This is interesting because they allow us to better realise a spatialisation increasing the sensation of naturalness. This idea can be implemented into multiple ways:

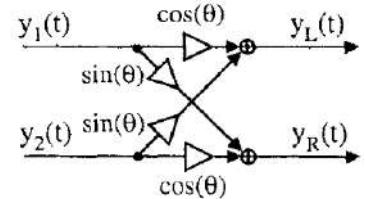
- We can have different outputs completely decorrelated one of the others
- We can multiply the output by a mixing matrix to correlate the outputs where the number of rows is equal to the number of comb filters, and the number of columns is equal to the number of outputs.

Schroeder suggested that the coefficients

of the mixing matrix have values of  $+1$  or  $-1$ , Jot suggested that the mixing matrix have orthogonal columns. Different mixing matrixes returns different spatialisation effects. For example, the following matrix produces a stereo reverberator that is quite spacious and enveloping when listened to over headphones.

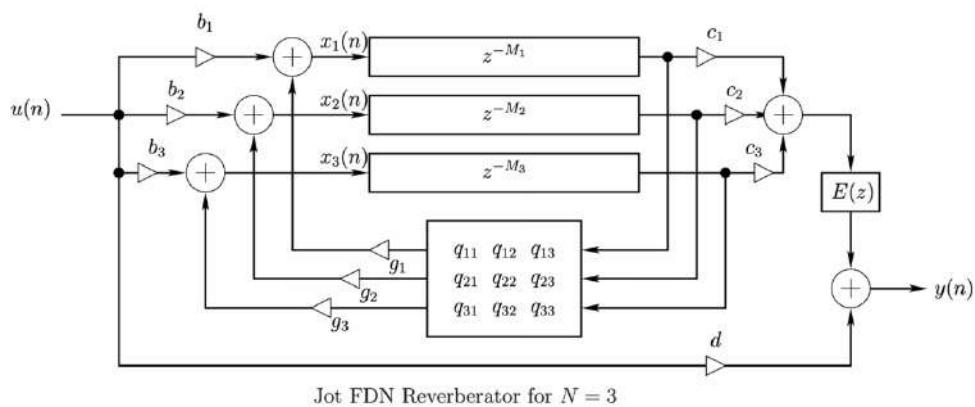
$$M = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Martin and Jot suggested that, given two mutually uncorrelated outputs  $y_1(t)$  and  $y_2(t)$ , we can mix them to achieve any desired amount of interaural cross-correlation as shown here.



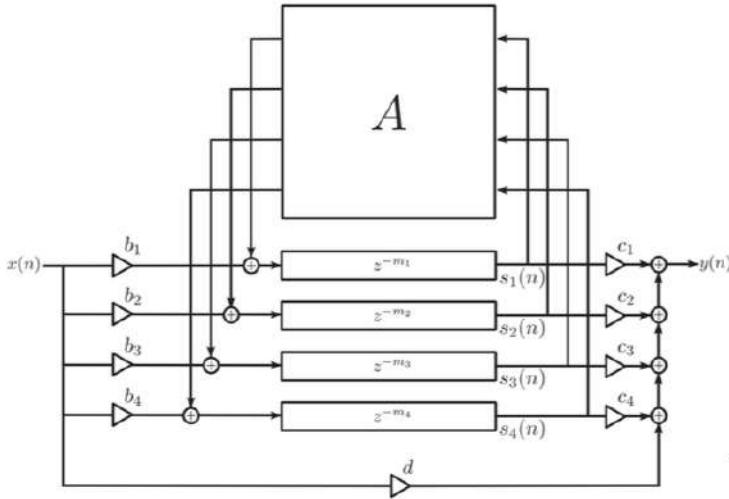
### Feedback Delay Networks

Let us see how to implement the feedback delay network, that is nothing but a generalisation of a comb filter. As we can see, there is a direct path weighted by  $d$  and a tonal correction filter called  $E(z)$  that equalises mode energy independently of reverberation time.



The mixing matrix multiply and sum different samples. It is an expanded and generalised version of comb filter. One important thing with this feedback delay network: we want the system to be lossless, the matrix must not kill frequencies. How can we do? The matrix should be a unitary matrix. Side note: it does not mean that the matrix must be a unitary matrix. The system surely works if the matrix is an unitary one, but there could be other type of matrixes for which it works.

Let us see a hint of what we can have by looking at this example in which we have a FDN (feedback delay network) with  $N = 4$ .



$$\underline{\underline{A}} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

$$y(n) = \sum_{i=1}^N c_i s_i(n) + d x(n)$$

$$s_i(n+m_i) = \sum_{j=1}^N a_{ij} s_j(n) + b_i x(n)$$

Let us re-write them in matrix fashion.

$$\underline{\underline{A}} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix} \quad \underline{\underline{b}} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} \quad \underline{\underline{c}} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} \quad \underline{\underline{s}}(n) = \begin{bmatrix} s_1(n) \\ s_2(n) \\ \vdots \\ s_N(n) \end{bmatrix}$$

$$\underline{\underline{D}_m} = \text{diag}([z^{-m_1}, z^{-m_2}, \dots, z^{-m_N}]) = \begin{bmatrix} z^{-m_1} & 0 & \cdots & 0 \\ 0 & z^{-m_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & z^{-m_N} \end{bmatrix}$$

The two starting equations becomes

$$\underline{\underline{S}}(z) = \underline{\underline{D}_m}(z) \left( \underline{\underline{A}} \underline{\underline{S}}(z) + \underline{\underline{b}} X(z) \right) \rightarrow \underline{\underline{D}_m}^{-1}(z) \underline{\underline{S}}(z) = \underline{\underline{A}} \underline{\underline{S}}(z) + \underline{\underline{b}} X(z)$$

$$\rightarrow \left( \underline{\underline{D}_m}^{-1}(z) - \underline{\underline{A}} \right) \underline{\underline{S}}(z) = \underline{\underline{b}} X(z) \rightarrow \underline{\underline{S}}(z) = \left( \underline{\underline{D}_m}^{-1}(z) - \underline{\underline{A}} \right)^{-1} \underline{\underline{b}} X(z)$$

$$Y(z) = \underline{\underline{c}}^T \underline{\underline{S}}(z) + d X(z) = \underline{\underline{c}}^T \left( \left( \underline{\underline{D}_m}^{-1}(z) - \underline{\underline{A}} \right)^{-1} \underline{\underline{b}} X(z) \right) + d X(z)$$

The transfer function will be therefore

$$H(z) = \frac{Y(z)}{X(z)} = \underline{\underline{c}}^T \left( \underline{\underline{D}_m}^{-1}(z) - \underline{\underline{A}} \right)^{-1} \underline{\underline{b}} + d$$

From this last equation we can compute which are the zeros and which are the poles and check the stability of our system.

The order of this system is equal to the sum of the delays and the zeros of its transfer function can be shown to be the roots of the polynomial

$$q(z) = \det \left[ \underline{D}_m^{-1}(z) - \underline{\underline{A}} + \frac{1}{d} \underline{b} \underline{c}^T \right]$$

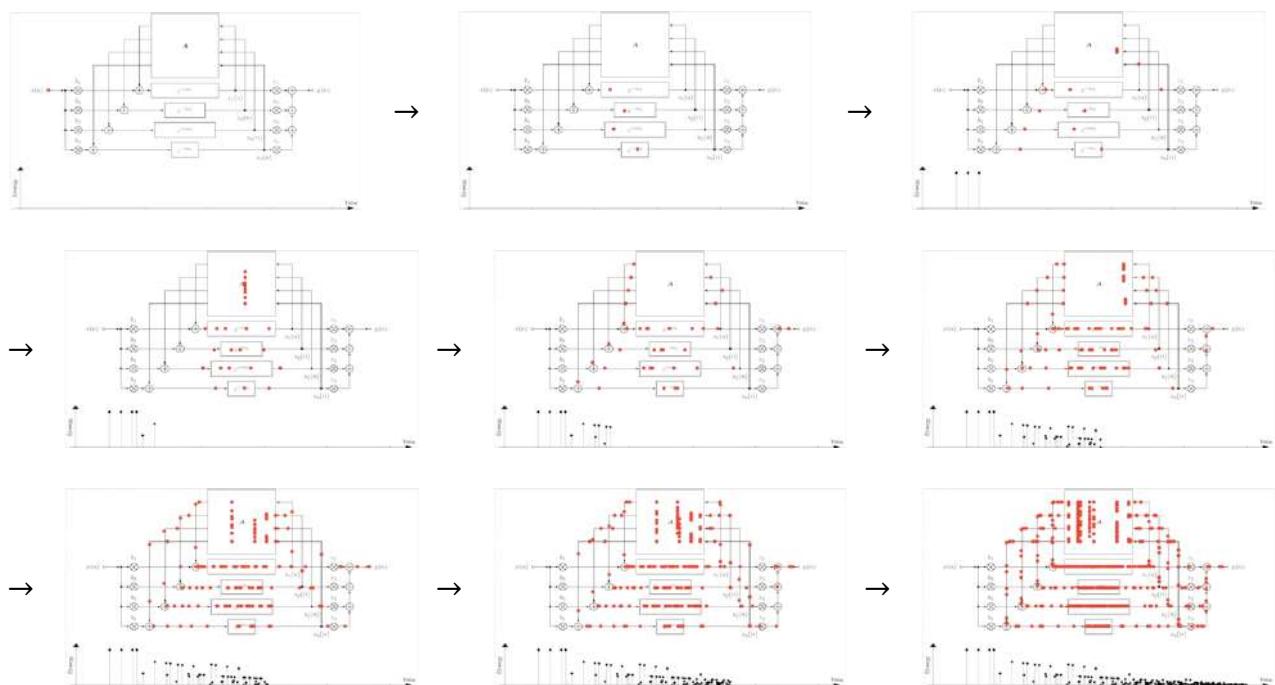
while the poles are the roots of the polynomial

$$p(z) = \det \left[ \underline{D}_m^{-1}(z) - \underline{\underline{A}} \right]$$

This last polynomial we have written is the generalised characteristic polynomial of  $\underline{\underline{A}}$  with delays  $\underline{m}$ .

Formally, we say that the FDN is lossless if  $p(z)$  has only “unimodular roots”, for example, if all of its roots have magnitude 1.

We report here, in a series of photograms, the interesting animation made by Sebastian Slecht in the AudioLabs of Fraunhofer, Erlangen.



What we see here is that when one sample enters in the feedback delay network, it enters in the several loops that are present in the scheme. Each time one is put out, a line appears on the energy-time graph. That is basically what happens when we create the impulse response of the system.

## Geometric reverberation Algorithms

We now deal with geometric acoustic methods. This means that to characterise a reverberation we have to be able to understand which are the paths from a source to a receiver. This is the same problem that we have in computer graphics with light. As we can model light, in the same way we can model sound reverberation. But which are the differences?



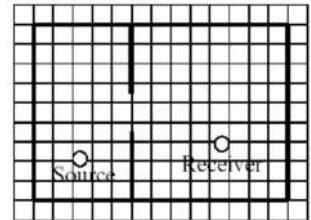
- Sounds has longer wavelength than lights: some effects like diffraction cannot be neglected: when a soundwave hits an object with comparable size it scatters all around.
- Soundwaves are coherent (modelling phase becomes important): this is the problem that appears when we have stereo systems at a concert, and we have to take into account interactions.
- Sounds travels more slowly than light: reverberations are perceived over time

### Approaches

There are five different kind of approaches we can have:

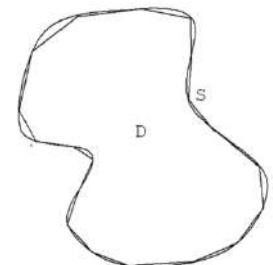
#### *Finite Difference Methods*

We describe the environment by using physics. The model is very precise and realistic. Furthermore, it is flexible: you are not bound by any constrain. Once the environment has been recreated, the wave equation is discretised over a grid-aligned mesh. As we can imagine, this is the finest model, but it is overly expensive.



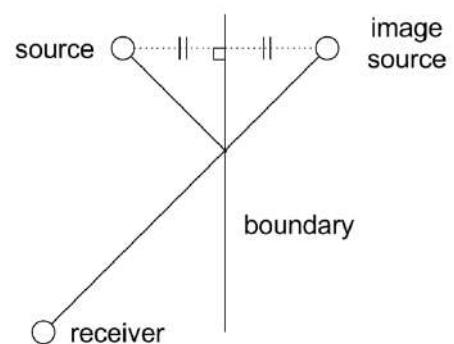
#### *Boundary Element Method*

Slightly better solution with something that takes physics into account but less expensive. The main difference between FDM and BEM is that we can exploit the fact that you can describe the wavefield in a point in space if you have information on the surface in which the point is. In fact, according to Kirchhoff Helmholtz's integral, we can represent what happens inside a volume by using a series of monopole-dipoles sources distributed along the surface of the volume itself. In this way we can discretise only the surface and make the computation easier. The formulation per se is simpler. We still inherit some of the disadvantages of the FDM: we have to discretise the source (lots of point to take into account since each element must be much smaller than the wavelength), form factors must model diffractions and specularities and so on. As, as we said, the elements must be much smaller than the wavelength, this method works well with low frequencies sound. Low frequency means large wavelength; the computation becomes lighter as we can discretise the surface with a larger pace.



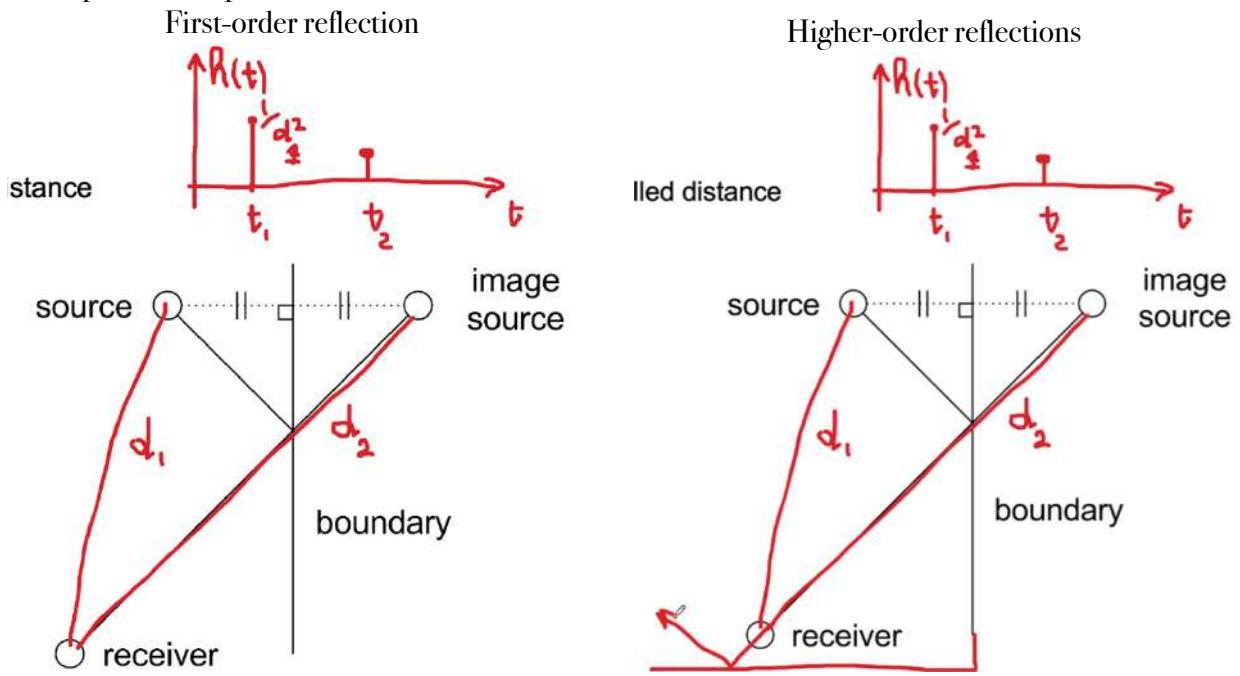
#### *Image Source Method*

Very simple method that can turn very useful. We suppose that sound propagates only along straight lines (rays), that sound energy travels at a fixed speed, that the energy decreases with  $\frac{1}{ray^2}$  and that the boundaries behave like perfect mirrors... as if we are playing pool. The single reflection gives the first-order reflection. Higher-order reflections are obtained through multiple mirror mappings about all involved boundaries. The RIR is found by adding the contributions of all image sources, delayed and attenuated based on the length of the travelled path. Simply and easy. Anyway, the



assumption that sound propagates only along straight lines is not so precise: outside there are lots of variation in the gradient of pressure and humidity and the sound does not propagate linearly.

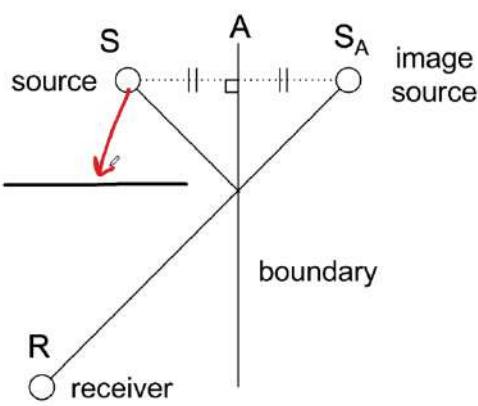
For example, in this specific case we have



Long story short for image source method is: we mirror the source and we evaluate the paths.

Let us suppose now to have another boundary at the back of our image: in order to understand the path, we mirror the sources once again as it is done in the picture above, on the right. And so on.

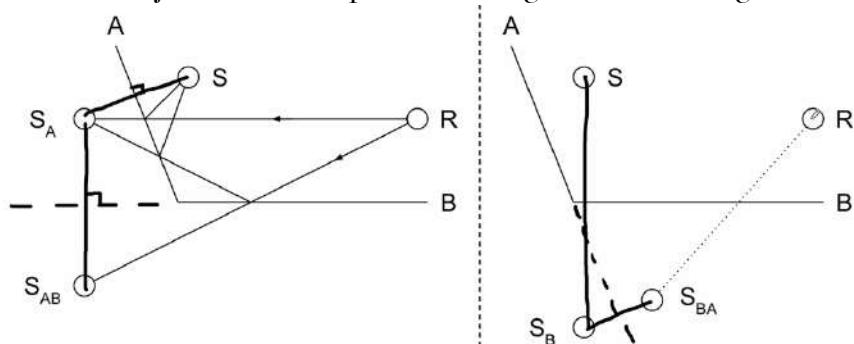
Couple of comments related to this: the higher the order that we consider, the more the image sources that we are creating. The number of sources scales very quickly with the order of the system. Furthermore, after some reflections, the sound is more diffused (scattered) than reflected making the validity of the mirror approximation expire. Another small issue is that not all paths are valid. For example, if we have a reflector between the source and the receiver, there is no direct path.



What we have to do to identify the proper paths is the audibility check: the path that I considered is realistic or not? Obviously, the check becomes more challenging as the numbers of sources increase. Let us consider the following example: we consider two walls,  $A$  and  $B$ . The image source  $S_{AB}$  has been reflected in  $A$  then in  $B$ . For the image source to be valid

- $R \rightarrow S_{AB}$  must intersect  $B$  at some point  $B_{int}$
- $B_{int} \rightarrow S_A$  must intersect  $A$  at  $A_{int}$
- $A_{int} \rightarrow S$  must not intersect with any scene geometry.

According to what we have just written, the path in the image below on the right is not realistic.

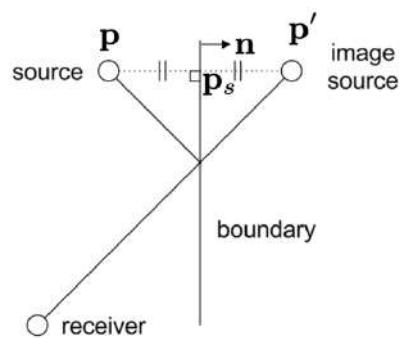


If we are not able to perform the mirroring visually, we can use the following equations.

For a point  $p$ , its reflection  $p'$  in a plane with unit normal vector  $n$  and intersecting the point  $p_s$  can be found by

$$p' = p - 2n(n \cdot (p - p_s))$$

If the surface is a mesh of triangles, the normal vector  $n$  is found by taking the cross-product of two of the triangle edge vectors, where an edge vector is the difference between two triangle vertices. The point on the plane  $t$  can be set to any of the triangle vertex positions.



There are some advantages for image source methods especially when we deal with rectangular rooms in which the results are quite acceptable even though we consider specular reflections only and visibility checks make the method exponentially (literally exponentially with the reflection order) expensive.

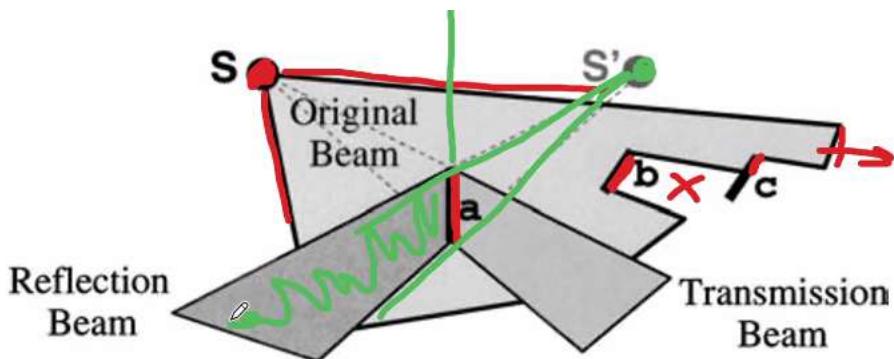
Stupid and quick question: if we consider only first order reflections, how many peaks do we expect in our impulse response? The first one + 6 (the four walls, floor and ceiling).

### *Path Tracing*

Brute force approach. We consider all the rays modelling all type of surfaces and scattering. With this kind of method, it is easier to implement scattering but, on the other hand, we cannot cast an infinite number of rays. We have to sample them but, by doing this, the risk is to lose the rays that are directed straight to the receiver.

### *Beam tracing*

Smarter solution. The idea is the same of beam tracing but rather than using rays we use beams (which are bundle of rays)

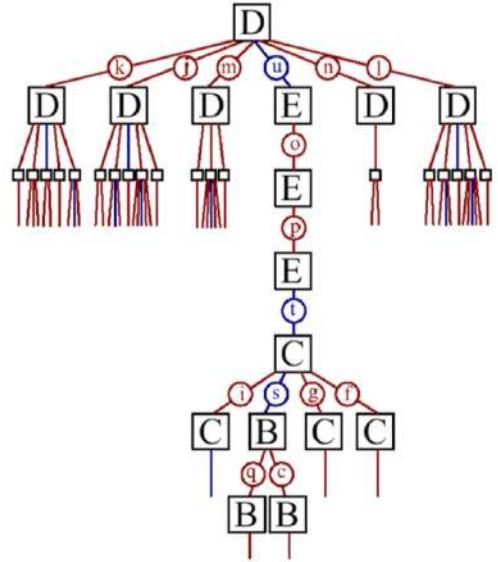
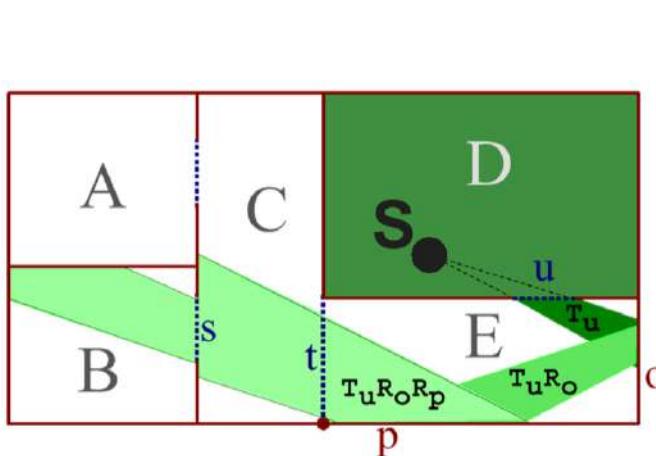


Let us suppose that the red source  $S$  emanates a beam. What we have to do is to check if there are reflector that are hit by the beam that “blocks” the beam (for example what it happens with  $b$  and  $c$ ). Anyway, the reflector also generates a reflection that can be described by another beam. It looks like the same we did for image source method but with beams instead of rays. Let us now consider the reflector  $a$ : it produces a reflected beam that is equal to the beam that the source  $S'$  (green), born from a mirroring of  $S$ , could produce. But for this last beam, the one generated by  $S'$ , we only consider the portion restricted to the corner of  $a$ . By iteratively considering beams reflected by reflectors, we are doing two interesting things:

1. We do not lose any ray: we are reasoning in terms of covered area; we do not miss a single point in that planar region. The problem of ray tracing is solved.
2. It is independent on the receiver. The receiver?

What about the receiver? We did not mention it at all. The visibility of the source is re-computed over the entire environment. To understand which is the impulse response of the room in a certain point, we simply consider the contributes of all the beams that are passing through that point. But how is done practically? Information is arranged in a tree fashion. Given the point, the contributes of the beams that point to that point can be evaluated by following the tree from the top.

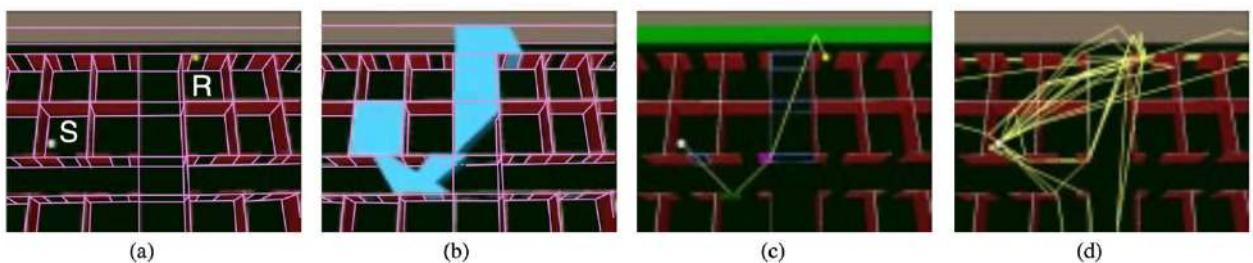
In the example below, the beam generated in the region  $D$  enters in the region  $E$  through the opening  $u$ . Then, it bounces on the surface  $o$  remaining in  $E$ . The same happens when it bounces on the surface  $p$ . After that, it enters in the region  $C$  through the opening  $t$  and, a portion of the same beam, enters in the region  $B$  through the opening  $s$ . This is the process with which the tree is populated. When we want to know which are the beams that are irradiated in a certain zone, we have to follow the tree backward.



Below an example that show the differences between beam tracing and ray tracing is reported.

Results of each phase of execution:

- Virtual environment (office cubicles) with source S, receiver R, and spatial subdivision (pink)
- Example reflected and diffracted beam (cyan) containing the receiver
- Path generated for the corresponding sequence of opaque faces (green), transparent faces (purple), and edges (magenta)
- Many paths found for different sequences from S to R



### Summary on geometric reverberation algorithms

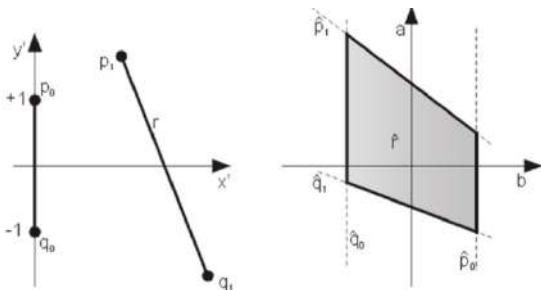
Let us summarise all the techniques

- FEM/BEM  
Best for low frequencies
- Image source methods  
Best for rectangular rooms (very common)
- Path tracing  
Best for high-order reflections (very common)
- Beam tracing (fast ray tracing)  
Best for precomputation, but only if the source is fixed, otherwise we have to recompute the beam tree (slow). It is, therefore, a fast ray tracer, but a slow beam tracer.
- Fast beam tracing (visibility lookup)  
Learning how to implement a fast beam tracer (fast computation of the beam tree) enables fast sound field rendering! The beam tree, in fact, contains all the information we need to compute the whole sound field

### Visibility region

There are ways to compute the visibility map (which tells us which reflector sees which reflectors) more quickly. In order to do so, we have to make some assumptions:

- Sources and Receivers are assumed to be point-like
- Reflectors are assumed to be oriented surface of a reflecting wall: a reflecting wall defines two flat reflectors identified by an index.

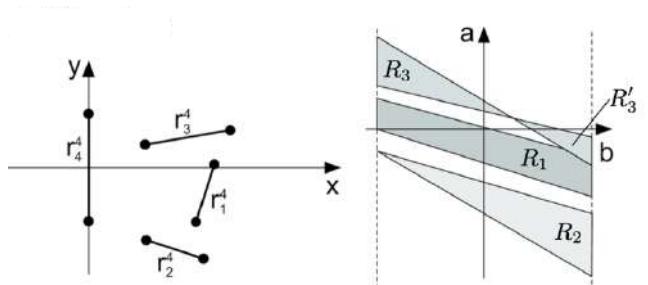


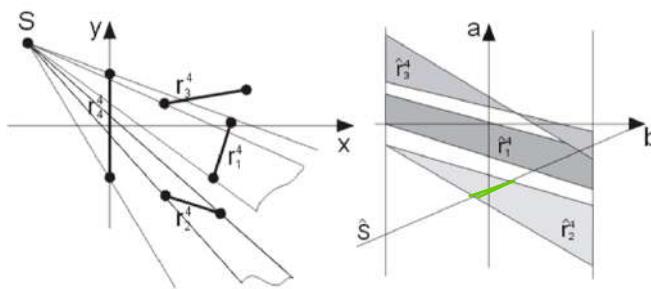
For example, let us consider the image on the left. How can we parametrise all possible rays that are generated behind the vertical line that lies on the  $y'$  axis and hit the other reflector (the tilted line)?

Being the reflector two lines, they will be parametrised by the angular coefficients  $a$  and the offset  $b$ .

$$y = ax + b$$

The combination or sub-combination of the rays generated by a source behind the vertical line defines all the possible beams that a source mirrored on the vertical line can generate. Since every ray (every line on the plane) is parametrised by  $a$  and  $b$ , a line is represented into a point in the  $(a, b)$  plane. The union of all possible lines passing through the vertical reflector and hitting the tilted one generate an area on the  $(a, b)$  space. So, that area will represent all possible beams that any source can generate considering this set. This is something that has to be computed for all the pair of reflectors that you have in a room. At the end, all the visibility regions are put together as it is done in the picture on the right. At a certain point we notice that two beams overlap. There, we have an issue that needs to be fixed by checking what actually happens.

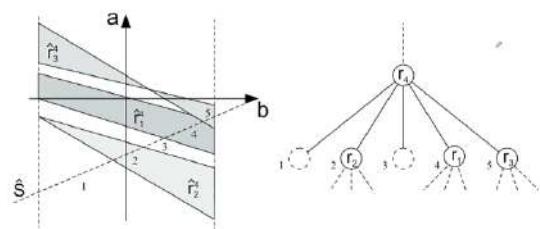




As a last step, we have to include the source in the parameters space. A source, on the  $(x', y')$  space is a point. If we use the coordinate of the point as parameter of a line, we can draw that line  $\hat{s}$  on the plane  $(a, b)$ .

$$b = -x_s a + y_s$$

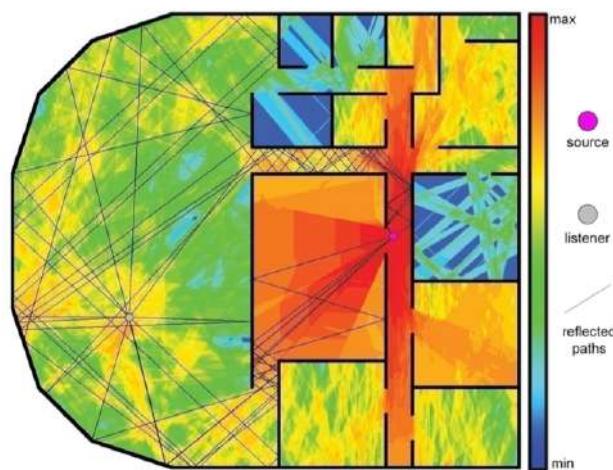
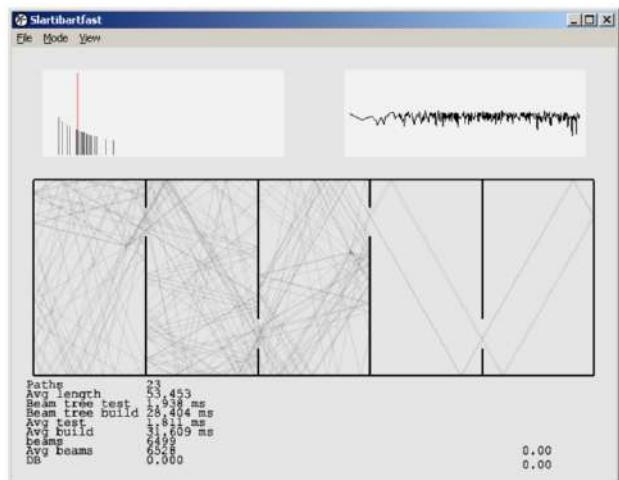
This is extremely cool because the interception between that line and the visibility region tells us which is the bundle of rays (the beam) from the source to the reflector that define that visibility region. All the pairs  $(a, b)$  on that line  $\hat{s}$  defines all the lines that goes from  $S$  passing through  $r'_4$  reflected by the reflector that labels the region on which the pair of the line is. For example, the points that from the green segment represent all the rays generated by  $S$  that pass through  $r'_4$  and hit  $r'_2$ . Once we defined the visibility regions, we populate the tree.



We can tell that this algorithm is pretty well optimised as it flies on a software for windows XP that implement it.

The average time for building the tree is just of 31.609 ms while the one needed for testing is of 1.811 ms. A great result.

Below, a picture of how a graphical interface of a software that computes the sound field by means of a beam tree looks like today.



I hope you find this notebook useful,  
Break a leg! ☺

### Exam:

- 5 question;
- 45 minutes for each module.

### Evaluation:

For both the modulus 20% of the grade comes from the homework while the other 80% from the exam. At the end will be evaluate the average between the two.

### Examples

**Q1.** You are developing a system for music transcription that works by tracking sinusoidal peaks in the time-frequency domain. You are tuning your system to work with a specific keyboard that reproduces harmonic sounds with fundamental frequency spanning the range 440-880 Hz (i.e., from A4 to A5). Consider that this keyboard is used to only play one key at a time (i.e., chords are not played).

- Sketch the spectrum obtained when playing the highest pitch note.
- Which is the minimum expected distance between two peaks in the frequency domain considering the described keyboard? Why?
- Considering a rectangular window (L factor equal to 2) and a sampling frequency of 44 kHz, which is the length (in seconds and samples) of the analysis window that allows you to resolve the sinusoidal peaks? Which is the condition related to the window main lobes to be satisfied in this situation?
- How would you change the analysis window if the keyboard fundamental frequency range changes to 440-1760 Hz?

**Q2.** Consider the problem of Time/Pitch Scaling

- What does it mean to apply Time Scaling to an audio signal?
- Consider that Time Scaling with factor 2 is applied to a 10-second long sinusoid with frequency 400Hz. What's the output of the Time Scaling operation?
- With the help of a block diagram, report the idea behind the duality principle.

**Q3.** Considering the problem of audio restoration, describe how the median filter works and explain under which conditions and for what kind of degradations it can be used. Is this a linear operation?

Not at all, it is a Wiener-Filtering

**Q4.** Consider Linear Predictive Coding (LPC)

- Describe the source-filter interpretation.
- Discuss why it is particularly suitable for modeling the human voice.
- Describe which kind of residual do you expect applying LPC to “voiced” and “unvoiced” signals.

For voiced signals train of pulses, for unvoiced something more noisy.

**Q5.** Consider the problem of far field source localization in case of narrow band signals

- Describe a Uniform Linear Array (ULA) and sketch its configuration.
- Report the condition to be fulfilled to avoid spatial aliasing.
- Show with a sketch how it is possible to link the Direction Of Arrival (DOA) to the Time Different Of Arrival (TDOA) measured between two microphones.

**Q6.** We want to determine the direction of arrival of a wideband source in an anechoic room. Report a possible solution to the problem, also considering the issues that may arise when periodic signals are considered.

This was exactly the Pezzoli's HW.

## Module 2

**Q1.** Consider the problem of sound synthesis by means of signal-based approaches

- Briefly report the idea behind the wavetable synthesis method
- Define and describe the Synchronous Overlap and Add (SOLA) method, explaining which is the wavetable synthesis problem that it can help solving.

**Q2.** Consider the problem of synthesizing a reverberated audio signal.

- Define the concept of Room Impulse Response (RIR) highlighting its component.
- Consider a rectangular room with one microphone and one source. Considering only first-order reflections (i.e., after one reflection, the signal does not “bounce” on walls anymore), sketch a possible RIR (hint: ignore the floor and the ceiling). Clearly report the labels on the axes.
- Explain how it is possible to use a RIR to apply reverberation in the digital domain to a dry sound recording.

## Convolution

**Q3.** Consider the implementation of audio effects (e.g., chorus, flanger, etc.) by means of delay lines.

- Describe the general concept of delay line applied to an audio signal.
- Describe the differences among: i) an integer delay line; ii) a fractional delay line; iii) a time-variant fractional delay line.
- Which kinds of delay lines need an interpolator? Why?

**Q4.** Consider the problem of binaural rendering.

- Describe what is the Head-Related Transfer Function (HRTF).
- Explain how it can be used for binaural rendering.
- Explain how it is possible to take into account a virtual environment.

You have also the reflections that has to be taken into account and be applied to the same dry signal.

**Q5.** Consider the WDF techniques.

- Briefly explain how to model a circuit using WDF, when the circuit has a resistive nonlinearity.
- How do you derive the nonlinearity in the wave digital domain starting from its Kirchhoff description?
- How does this wave description of the nonlinearity depend on the rest of the circuit?

**Q6.** Consider Wave Field Synthesis technique.

- Describe the main idea behind it, and its relation to the Kirchhoff-Helmholtz integral.
- What are the main limitations when it comes to implementing Wave Field Synthesis in practice?

Set of secondary sources modelled using the Huygens principle

In second module it is less likely to have numerical questions

# SAMPLE QUESTIONS FOR THE EXAM OF SOUND ANALYSIS, SYNTHESIS AND PROCESSING

## DIGITAL AUDIO ANALYSIS AND PROCESSING (DAAP)

Below you can find a set of open questions that can be asked for the exam of the first module.

- Explain how to choose a window in order to guarantee that you will be able to discriminate two spectral peaks that are  $f_0$  Hz apart using a STFT.
- When and why do we need zero-padding in implementing a STFT-based analysis scheme? Please explain.
- Define Time Scaling for periodic signals, write the formula and motivate it.
- Explain how to implement Time Warping (resampling) in the frequency domain (using the STFT). Discuss an alternative method in the time domain and explain when it would be recommendable to opt for it instead of working in the frequency domain.
- Describe the Generalized Cross Correlation method for the localization of wideband sources. In particular, explain why we should adopt a pre-whitening filter for the signals.
- Describe the similarity between spatial filtering and temporal filtering. Based on this similarity, derive the condition on the minimum distance between adjacent microphones.
- You need to restore an audio signal corrupted by clicks, stationary background noise and template pulse noises. Describe, with the help of a block diagram, the whole restoration processing, and make sure you specify the order in which the individual restoration operations are to be performed. Briefly describe the various blocks and justify the answer.
  
- Discuss the parametric Wiener Filter in the frequency domain, and explain which parameters of this filter lead to Power Spectral Matching (justify the answer in formulas).
- With reference to Linear Predictive Coding (LPC), explain the meaning of the terms "whitening filter" and "shaping filter" (we suggest to refer to a predictor of infinite order).

The level of formalism: the Wiener filtering and noise removal and restoration requires formal answers.

## SOUND SYNTHESIS AND SPATIAL PROCESSING (SSSP)

Below you can find a set of questions that can be asked for the exam of the second module.

- Discuss why and in what cases we need to use a filter approximating a fractional delay. Describe the frequency and impulse response of the ideal fractional delay filter.
- Briefly describe the derivation of Digital WaveGuide structures starting from the D'Alambert Equation.
- What is the function of a two-port junction in the Wave Digital domain? Explain the differences between a scattering junction for Digital WaveGuide structures and a Kelly-Lochbaum Junction for Wave Digital Filters as far as their interpretation is concerned.
- Discuss the primary auditory cues for the perception of direction in free field. What is the effect of head movement on these cues? What do you need to consider when considering primary auditory cues in a reverberant environment?

- Describe the basic principles behind the formulation of Higher-Order Ambisonics in 2D, with particular emphasis on the physical principles supporting the method. Discuss the relationship between the number of loudspeakers, the size of the reproduction area and the frequency range of operation.
- Discuss the primary auditory cues for the perception of direction in free field. What is the effect of head movement on these cues? What do you need to consider when considering primary auditory cues in a reverberant environment?
- Define the clarity index in the description of reverberation and explain why it is defined that way.
- Describe the differences between the following three kinds of representation of the sound field used both in the recording phase of a sound scene and in the reproduction phase: channel-based approach, transform-domain approach, and object-based approach.
- Discuss the Vector-Based Amplitude Panning method for multichannel stereophony. Determine the panning functions. What are the advantages with respect to two channel stereophony?
- Describe the concept of room impulse response (RIR) and its use in audio reverberation.

#### Next year

- Mirko Pezzoli will be the teacher of the whole course. Since there will be no continuity, they suggest us to do everything this year.