

Advanced Signal Processing

(Course materials)

Master 1 : Telecommunication & Networks

Faculty of Electrical Engineering

Houari Boumediene University of Science and Technology

Personal website: <http://perso.usthb.dz/~akourgli/>

e-mails: akourgli@usthb.dz , kourgliassia@gmail.com

Course's content

Introduction: Miscellaneous Reminders

Chapter 1. Reminders on digital filters (FIR and IIR) (3 Weeks)

1. Transformed in Z
2. Structures , transfer functions , stability and implementation of digital filters (FIR and IIR)
3. Digital minimum phase filter
4. Synthesis methods of FIR filters and IIR filters
5. Multirate digital filters

Laboratory Work n°1: Synthesis of FIR and IIR filters

Laboratory Work n°2: Multi -rate filtering

Chapter 2. Random Signals and Stochastic Processes (4 Weeks)

1. Reminder about Random Processes
2. Notions of stochastic processes
3. Stationarities in the broad and strict sense and Ergodicity
4. Examples of stochastic processes (Poisson process, Gaussian and Markovian process)
5. Higher order statistics (Moments and cumulants , Polyspectra , non-Gaussian processes, nonlinear treatments)
6. Power Spectral Density
7. Matched filter , Wiener filter.
8. Periodogram, correlogram, averaged periodogram , smoothed periodogram
9. Introduction to Particle Filtering

Laboratory Work n°3: Random Processes, periodogram and variants

Laboratory Work n°4: Matching filtering and Wiener filtering

Adaptive Digital Filtering (4 Weeks)

1. Parametric Methods
2. AR model (Lévinson , Yulewalker , Burg, Pisarenko , Music ...)- ARMA model
3. LMS Stochastic Gradient Algorithm
4. RLS Recursive Least Squares Algorithm

Laboratory Work n°5: AR modeling and adaptive filtering (LMS)

Chapter 4. Time-Frequency and Time-Scale Analysis (4 Weeks)

1. Time-frequency duality
2. Short term Fourier transform
3. Continuous, discrete and dyadic wavelets
4. Multi-resolution analysis and wavelet bases
5. Wigner-Ville transform
6. Time-Scale Analysis ,

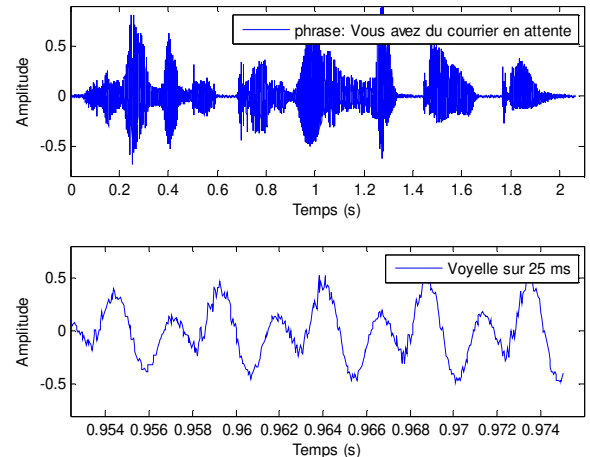
Laboratory Work n°6: Time- frequency /Scale transforms

Introduction: Miscellaneous Reminders

A **signal** is the physical representation of the information it carries from its source to its recipient. It serves as a vector for information. It constitutes the physical manifestation of a measurable quantity (current, voltage, force, temperature, pressure, etc.). The signals are electrical quantities varying as a function of time $x(t)$ obtained using sensors. Analytically, a signal will be a function of a real variable, generally time.

Examples:

- Acoustic wave: delivered by a microphone (speech, music, etc.)
- Biological signals: EEG, ECG
- Voltage at the electronic component terminals
- Geophysical signals: seismic vibrations
- Finances: oil prices
- Pictures, Videos



Note : Any physical signal has a random *component* (external disturbance, noise, measurement error, etc.). **Noise** is defined as any disturbing phenomenon that interferes with the perception or interpretation of a signal, by analogy with acoustic nuisances (interference, background noise, etc.). The differentiation between signal and noise is artificial and depends on the interest of the user: electromagnetic waves of galactic origin are noise for a satellite telecommunications engineer and a useful signal for radio astronomers.

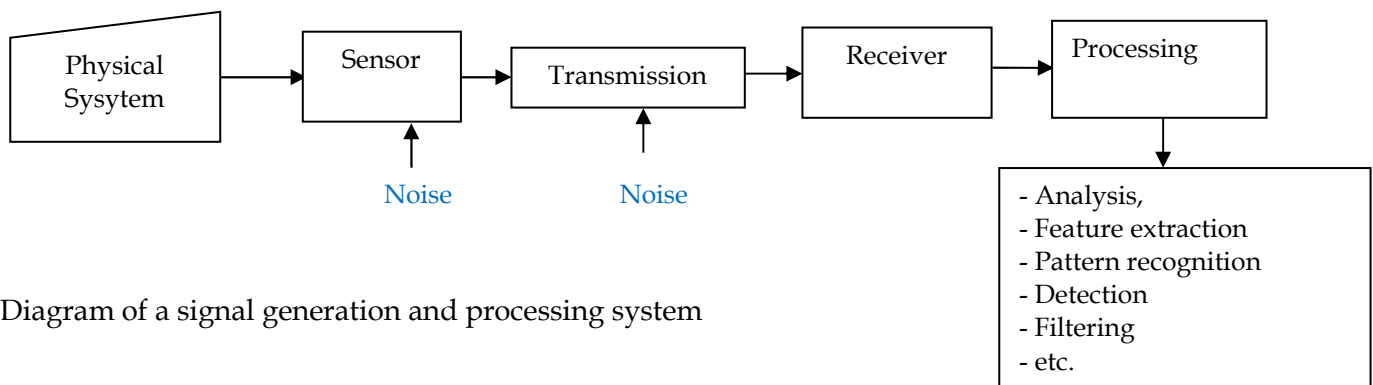


Diagram of a signal generation and processing system

Signal processing functions can be divided into two categories: signal processing (information incorporation) and signal interpretation (information extraction). The main functions integrated in these two parts are the following [1]:

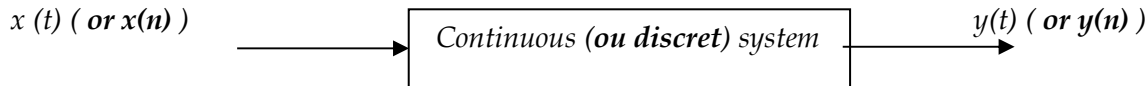
Signal processing: synthesis, modulation, coding/compression, etc.

Signal interpretation : filtering, detection, identification, analysis, measurement, etc.

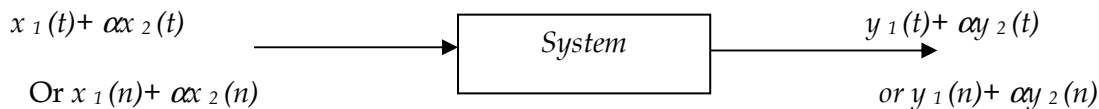
1. Theory of discrete and continuous linear and time-invariant (LTI) systems

A linear system is a system model that applies a linear operator to an input signal. It is a very useful mathematical abstraction in automation, signal processing, mechanics and telecommunications. Linear systems are thus frequently used to describe a nonlinear system ignoring small nonlinearities.

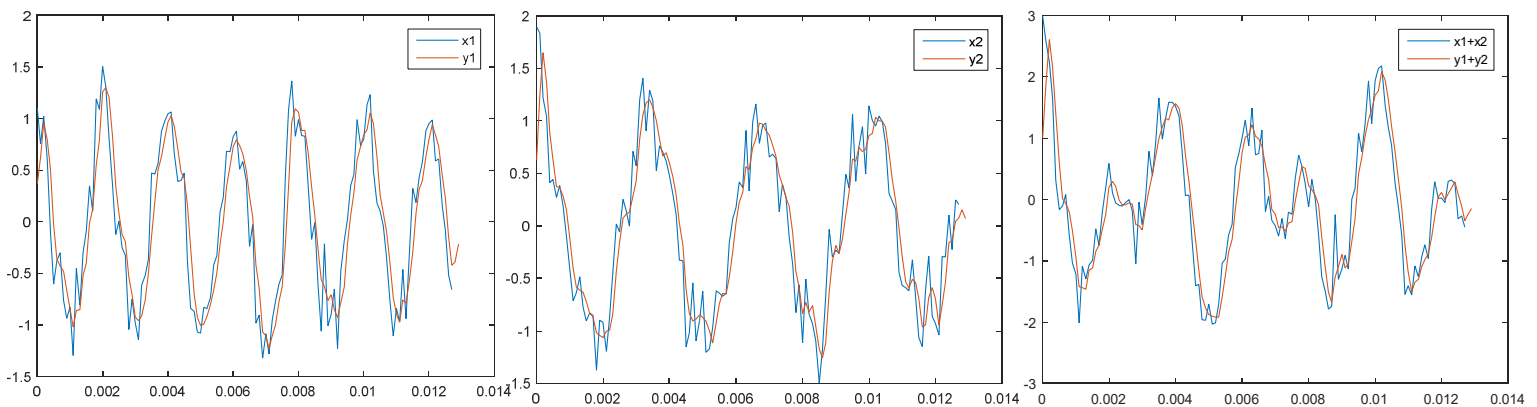
A system is continuous if at a continuous input $x(t)$, it provides a continuous output $y(t)$. A discrete system will make correspond to a sequence of discrete inputs $x(n)$ a sequence of discrete outputs $y(n)$.



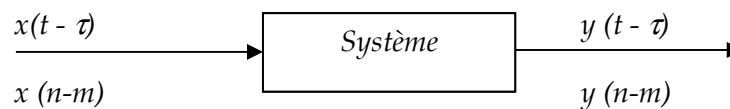
- The system will be said to be linear if when we apply an input $kx(t)$ (or $kx(n)$), the output will be $ky(t)$ (or $ky(n)$). If two inputs $x_1(t)$ and $x_2(t)$ generate two outputs $y_1(t)$ and $y_2(t)$ then $x_1(t) + x_2(t)$ will generate $y_1(t) + y_2(t)$. (Similarly, in the discrete case: if two inputs $x_1(n)$ and $x_2(n)$ generate two outputs $y_1(n)$ and $y_2(n)$ then $x_1(n) + x_2(n)$ will generate $y_1(n) + y_2(n)$)



Example

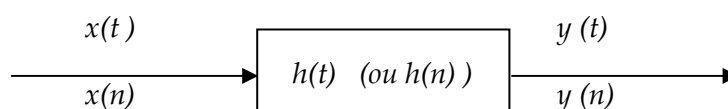


- If time is invariant, a translation of the input $x(t) \Rightarrow x(t - \tau)$ (ou $x(n) \Rightarrow x(n - m)$) will result in the same translation in time of the output $y(t) \Rightarrow y(t - \tau)$. (ou $y(n) \Rightarrow y(n - m)$).



If the system is invariant, this implies that the system reacts in the same way, whatever the instant at which we apply its excitations. This property expresses that the characteristic of the system does not depend on the origin of time, we still speak of stationarity.

Convolution : If the hypotheses of linearity and time invariance are verified, the system can be characterized by its impulse response $h(t)$ (or $h(n)$).



We can deduce the effect of any input in the form of a convolution. The latter is the most fundamental signal processing operation. It indicates that the value of the output signal at time t (or n) is obtained by the weighted (integral) summation of the past values of the excitation signal $x(t)$ (or $x(n)$). The weighting function is precisely the impulse response $h(t)$ (or $h(n)$) :

$$y(t) = x(t) * h(t) = \int x(\tau)h(t - \tau)d\tau = \int x(t - \tau)h(\tau)d\tau$$

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

The impulse response $h(t)$ (or $h(n)$) is the signal obtained at output $y(t)=h(t)$ (or $y(n)=h(n)$) if we apply a "Dirac's" $x(t)=\delta(t)$ (or $x(n)=\delta(n)$). Thus, the Dirac is the neutral element of the convolution operation:

$$x(t) * \delta(t) = x(t)$$

$$\delta(n) * x(n) = x(n)$$

Some properties of Dirac

$$\int_{-\infty}^{+\infty} x(t)\delta(t - t_0)dt = x(t_0)$$

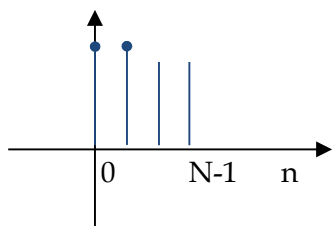
$$x(t) * \delta(t - t_0) = x(t - t_0)$$

$$x(t)\delta(t - t_0) = x(t_0)\delta(t - t_0)$$

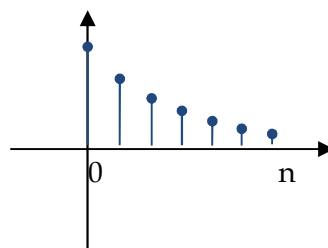
The calculation of the convolution therefore consists of calculating the sum of the product $x(\tau)h(t - \tau)$ (or $x(m)h(n - m)$). The signal $h(t - \tau)$ (or $h(n - m)$) is simply the initial signal $h(\tau)$ (or $h(m)$), reversed in time to give $h(-\tau)$ (or $h(-m)$) then translated of t (or n). By then calculating the set of products obtained by "dragging" h , i.e. for all shifts of t (or n), we obtain the convolution product for all t (or n).

Example 1 : (discrete)

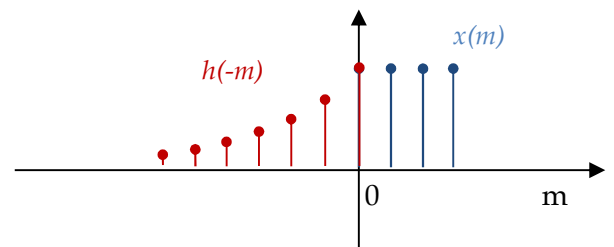
$$x(n) = \text{rect}_N(n)$$



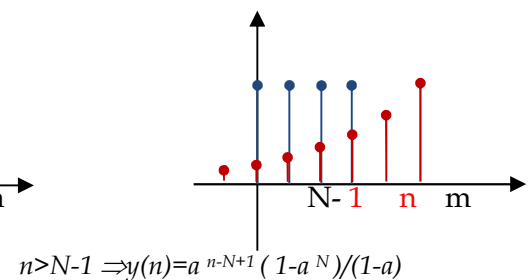
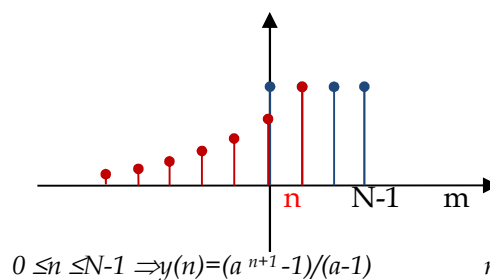
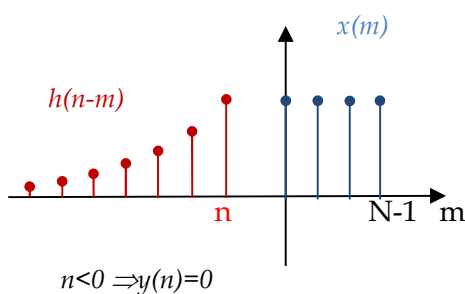
$$h(n) = a^n U(n) \quad 0 < a < 1$$



$$y(n) = x(n) * h(n)$$

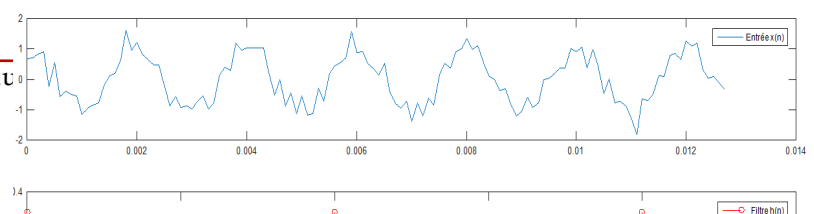


We distinguish 3 cases:



Example 2 : (See LW n°1)

FGE, USTHB [assiakourgli@gmail.com / <http://perso.u>



$$h(t) = \frac{1}{T} \Pi\left(\frac{t}{T}\right) \Rightarrow y(t) = \frac{1}{T} \int_{t-T/2}^{t+T/2} x(\tau) d\tau$$

$$h(n) = \frac{1}{N+1} \Pi\left(\frac{n}{N+1}\right)$$

$$\Rightarrow y(n) = \frac{1}{N+1} \sum_{m=-N/2}^{N/2} x(n+m)$$

Example 3: Let the signal $x(n) = \{2, -1, 3\}$ and $h(n) = \{1, 2, 2, 3\}$

Calculate $y(n) = x(n) * h(n)$

For finite sequences, one can use the column method

$$Y(n) = \{2, 3, 5, 10, 3, 9\}$$

$h[n]$	=	1	2	2	3
$x[n]$	=	2	-1	3	
		2	4	4	6
			-1	-2	-3
				3	6
					6
					9
		2	3	5	10
					3
					9

Remarks

When applying a sinusoid to a LTI, then the output will be a sinusoid whose amplitude and phase can be modified, but it will keep the same shape (a sinusoid) and the same frequency f_0 . We say that the sinusoids are the eigenfunctions of the LTIs.

An invariant linear system is a system whose behavior over time can be described by:

- either by a linear differential equation with constant coefficients: $\sum_{i=0}^M a_i \frac{d^i y(t)}{dt^i} = \sum_{i=0}^N b_i \frac{d^i x(t)}{dt^i}$

- either by a difference equation, for the discrete case: $\sum_{i=0}^M a_i y(n-i) = \sum_{i=0}^N b_i x(n-i)$,

2. Stability and causality

An important constraint for the formalization of many problems is to respect the notion of *causality* (the effects cannot precede the cause). In the case of LTIs, this causality results in the fact that for: $h(t) = 0$ for $t < 0$ (or $h(n) = 0$ for $n < 0$).

$$x(t) = 0, t < t_0 \text{ then } y(t) = 0, t < t_0 \Rightarrow h(t) = 0, t < 0, y(t) = \int_{-\infty}^t x(\tau) h(t-\tau) d\tau = \int_0^{+\infty} x(t-\tau) h(\tau) d\tau$$

Which gives us discretely:

$$x(n) = 0, n < n_0 \text{ then } y(n) = 0, n < n_0 \Rightarrow h(n) = 0, n < 0,$$

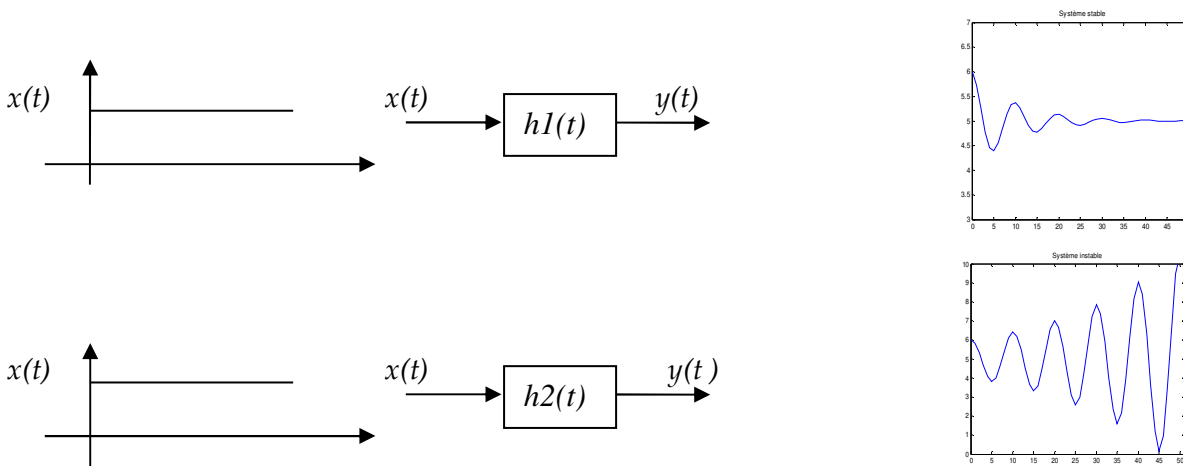
$$y(n) = \sum_{m=-\infty}^n x(m) h(n-m) = \sum_{m=0}^{+\infty} x(n-m) h(m)$$

- if h and x are causal $y(n) = \sum_{m=0}^n h(n-m) x(m)$

Note :

We can consider memorizing the input signals and processing them in deferred time, the systems used are then no longer necessarily causal because to produce the output at time t_i (or n_i), we have memory of the entries at the following instants. This is often the case in image processing, in speech processing performed after storing the signal to be processed.

Another fundamental notion is the *stability* of systems. The stability property of looped systems is not only a performance but a requirement for the proper functioning of a servo or regulation loop. An unstable loop is an unusable loop. The most common definition of this stability is as follows:



A system is said to be stable if, by applying any bounded input to it, the output remains bounded, which implies in the case of LTIs:

$$\int_{-\infty}^{+\infty} |h(t)| dt < M \quad \text{Or} \quad \sum_n |h(n)| < \infty$$

Note : For the rest, the definitions will be given for the discrete case.

3. Energy and power

Any transmission of information is accompanied by energy transfers. Indeed, continuous or discrete signals are essentially characterized by the energy or power they carry. These are the only physical quantities to which the detectors are sensitive. Many physical sensors measure energy or a quadratic quantity. For example, optical sensors measure intensity, electricity meters measure energy, etc. Taking into account the fundamental definition, the energy of the signal between times t and $t+dt$ is: $|x(t)|^2 dt$ (instantaneous power multiplied by time).

$$\sum_{-\infty}^{+\infty} |x(n)|^2$$

Let $x(n)$ be a discrete-time signal, such that exists and converges. Then the signal is said to have finite energy and the value of this sum is called the energy of the signal:

$$E_x = \sum_{-\infty}^{+\infty} |x(n)|^2$$

Examples:

$x(n) = \text{Rect}(n/N)$ finite energy. $x(n) = a$ (constant) and $x(n) = A \sin(2\pi f_0 n)$ are not finite energy

For a periodic signal, this sum does not converge. We can nevertheless define the power of a periodic signal $x(n)$ of period N by:

$$P_x = \frac{1}{N} \sum_{-N/2}^{N/2-1} |x(n)|^2 \quad \text{Or} \quad P_x = \frac{1}{2N} \sum_{-N}^{N-1} |x(n)|^2$$

In the general case, we speak of finite average power signals defined by:

$$P_x = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{-N/2}^{N/2-1} |x(n)|^2 \quad \text{Or} \quad P_x = \lim_{N \rightarrow \infty} \frac{1}{2N} \sum_{-N}^{N-1} |x(n)|^2$$

Examples :

Continuous signal $x(t)=a$, $A \sin(2\pi f_0 t)$, periodic signals, unit step, Dirac comb.

There are signals that are neither periodic nor of finite energy, for which the power cannot be defined, such as for example the ramp $x(n)=n$. These are mathematical definitions, in practice a measured signal is never measured over an infinite time interval. We can begin to visualize a signal at an instant that we will take as the origin of the times, and in this case we will stop its examination after a time T_{obs} :

$$E_x = \sum_{n=0}^{N_{obs}} |x(n)|^2$$

Noticed

Zero power finite energy signal \Rightarrow

Finite power signal \Rightarrow infinite energy

Calculating the energy or power makes it possible to obtain an initial characterization of the signal. In addition, signal theory has largely developed study methods based on correlation to characterize the temporal behavior of the signal.

Application exercise :

Calculate energy and power of signals: $\Pi_7(n)$, $A \cos(2\pi f_0 n)$.

4. Correlation and auto-correlation

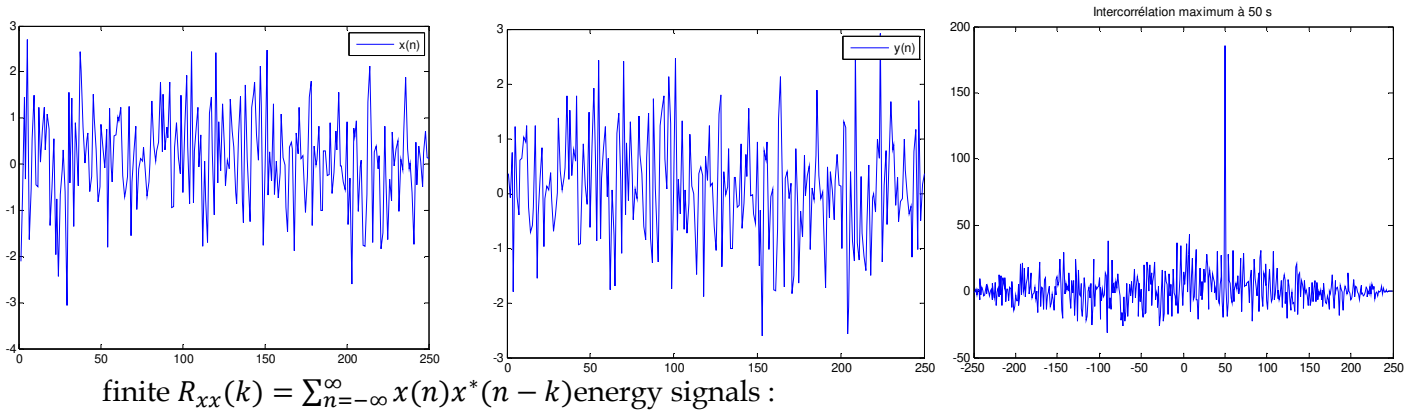
The correlation function makes it possible to measure the degree of resemblance between two signals as a function of a shift. Consider $x(n)$ and $y(n)$ two finite energy signals, the cross-correlation function $R_{x,y}(k)$ is defined by: $R_{x,y}(k) = \sum_{n=-\infty}^{\infty} x(n)y^*(n-k)$

The cross-correlation between $x(t)$ and $y(t)$ reaches a maximum for a lag k if $x(n)=y(nk)$

For finite average power signals, it is: $R_{x,y}(k) = \frac{1}{N} \sum_{n=-N/2}^{N/2-1} x(n)y^*(n-k)$

Examples

Consider a random signal and its version shifted by 50s. We notice that the signals resemble each other the most when $y(n)$ is shifted by 50 seconds.



Auto-correlation makes it possible to detect regularities, repeated profiles in a signal such as a periodic signal disturbed by a lot of noise (See LW n°1)

Properties :

- For $k=0$, we find the energy of the signal $R_{xx}(0) = E_x$ and $R_{xx}(k)$ is maximum at $k=0$
- If $x(n)$ is real, the auto-correlation is real and even.
- The auto-correlation of a signal of duration N will have a size $2*N-1$

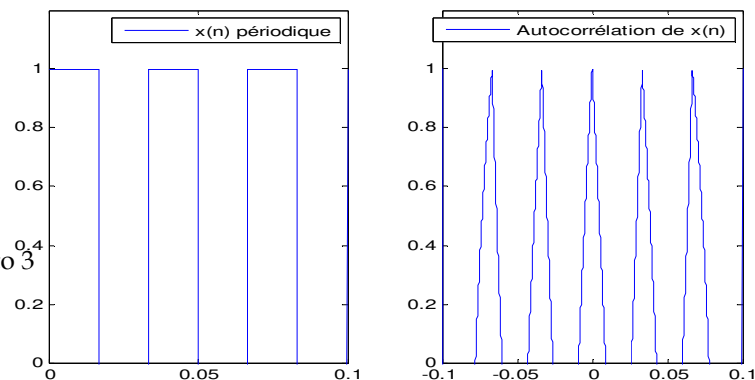
Auto-correlation of periodic signals: The calculation over a single period is sufficient. The auto-correlation of a periodic signal is itself periodic. By definition, the periodic signal perfectly resembles itself, shifted by one or more periods.

- periodic signals

$$R_x(k) = \frac{1}{N} \sum_{n=-N/2}^{N/2-1} x(n) x^*(n-k)$$

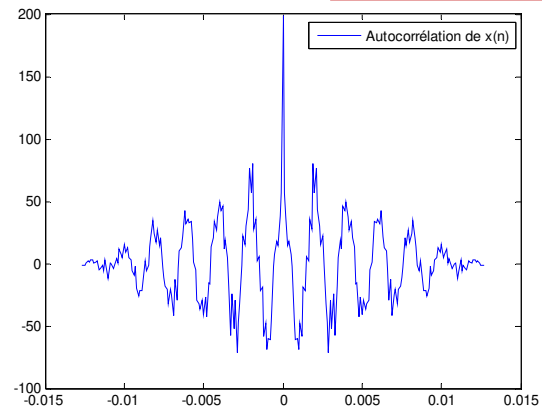
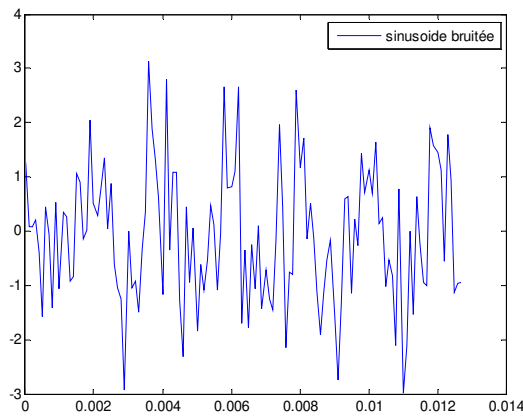
Application exercise Consider the signal $x(n)=(n+1)$ for $n=0$ to 3

Calculate the autocorrelation of x and deduce its energy

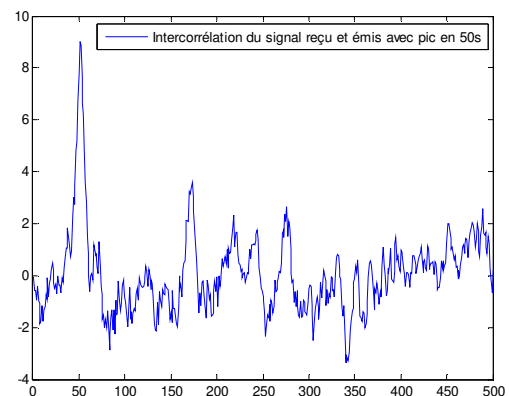
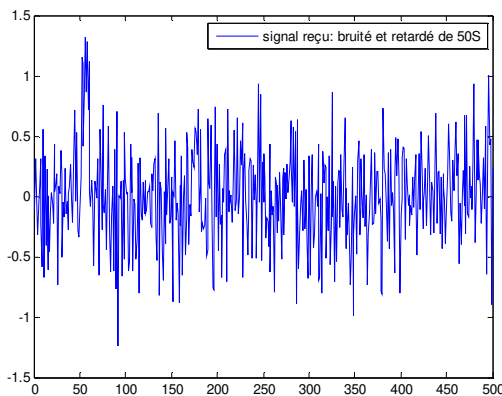
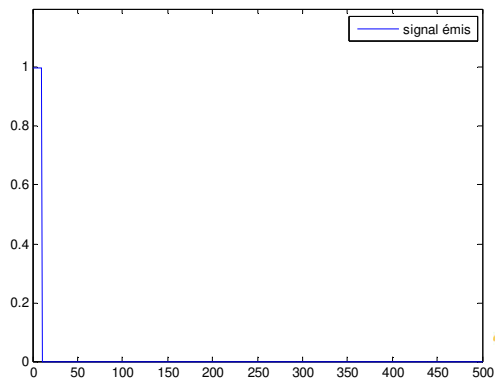


5. Fundamental applications of correlation functions

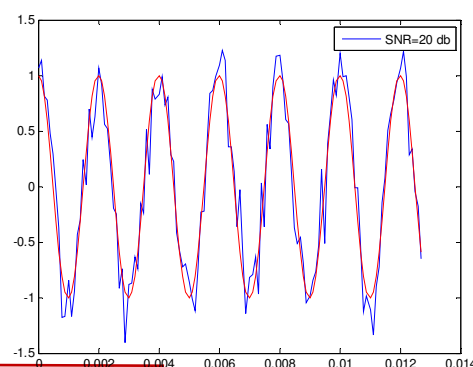
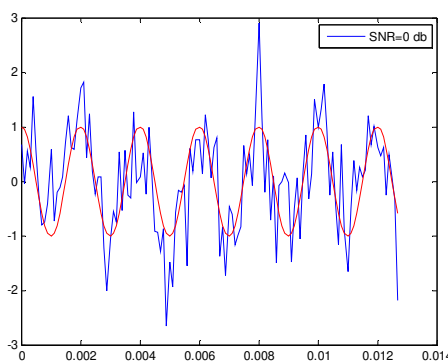
Extraction of a signal buried in noise, measurement of a time or delay, detection of a periodic signal (See LW no. 1) . The example below illustrates the autocorrelation of a sinusoidal signal of amplitude 1 buried in Gaussian noise of variance 1.



Correlation is widely used in radar systems. So, to detect an aircraft, we send a pulse, then we receive a delayed, attenuated and noisy version of this pulse. The cross-correlation of the received and transmitted signal will present a peak at the time corresponding to the delay.



Note: The notion of noise is relative, it depends on the context. The signal-to-noise ratio designates the quality of the transmission of information in relation to interference. It is defined by: $SNR_{db} = 20 \log(P_S / P_B)$



6. Finite or Infinite Impulse Response Filters (FIR and IIR)

- If the a_i are $\neq 0$, the system is said to be recursive (IIR), it is non-recursive if it only depends on the $x(n)$ (FIR)

- If the system is finite duration impulse response (FIR), then:

$$y(n) = \sum_{m=0}^K h(m)x(n-m)$$

In this case, the digital system is a window centered on the K most recent samples.

- If the system is an impulse response of infinite duration (IIR):

$$y(n) = \sum_{m=0}^{+\infty} h(m)x(n-m)$$

In this case, it is necessary to know all the present and past samples, the system has a memory of infinite length.

Example 1 $y(n)=x(n)+a_1x(n-1)+a_2x(n-2)+\dots+a_kx(n-k)$ is the finite difference equation for a FIR filter with the impulse response $h(n)=\delta(n)+a_1\delta(n-1)+a_2\delta(n-2)+\dots+a_k\delta(n-k)$ which, as we can see, is well finished.

Example 2 $y(n)=x(n)+a_1y(n-1)$ is the recursive difference equation of an IIR filter

with $y(n-1)=x(n-1)+a_1y(n-2) \Rightarrow y(n)=x(n)+a_1x(n-1)+a_1^2y(n-2)$

similarly $y(n-2)=x(n-2)+a_1y(n-3) \Rightarrow y(n)=x(n)+a_1x(n-1)+a_1^2x(n-2))+a_1^3y(n-3)$

$\Rightarrow y(n)=x(n)+a_1x(n-1)+a_1^2x(n-2))+a_1^3x(n-3)+a_1^4x(n-3)+\dots+a_1^m y(nm)$

Continuing the process ad infinitum $y(n)$ depends on an infinity of $x(nk)$ which makes it an IIR filter.

Application example

The sequences $x(n)$ (real) and $y(n)$ represent the input and output of a discrete system, respectively. For each case, identify those representing

- a) linear systems, b) causal systems, c) invariant systems to translations of n ,
d) definitely or possibly stable systems (depending on the constants)

1. $y(n) = x(n) + bx(n-1)$ 2. $y(n) = x(n) + bx(n+1)$ 6. $y(n) = b^{x(n)}$ b : real constant
3. $y(n) = nx(n)$ 5. $y(n) = x(n)e^n$ 7. $y(n) = |x(n)|$
4. $y(n) = x(n) \sin(2\pi f_0 n)$

- a) All except 6 and 7 b) All except 2 c) Systems 1, 2, 6 and 7 d) 1 (finite b), 2 (finite b), 4, 6 (finite b) and 7.

I. Reminders on digital filters (FIR and IIR)

The Fourier transform is a valuable tool for analyzing and processing signals. However, in certain problems (such as digital filtering), the limits of TF are quickly reached. The Z transform, which applies to discrete signals, generalizes the TF and allows these limits to be overcome [10]. It is completely analogous to the Laplace transform, but easier to use. This type of transform makes it possible to easily describe signals at discrete time and the response of invariant linear systems subjected to various inputs. It is a tool that makes it possible to calculate the impulse response of an invariant linear system described by a finite difference equation. It allows direct interpretation of signal and filter characteristics in the frequency domain [9].

1. Z-transform and properties

- **Definition :** The ZT is the generalization of the DTFT ($X(f) = \sum_{n=-\infty}^{+\infty} x(nT_e)e^{-2\pi j\vec{r}fnT_e}$).

Consider a discrete signal $x(n)$. Its ZT is defined by:

$$X(z) = \sum_{n=-\infty}^{+\infty} x(n).z^{-n} \quad \text{where } z \text{ is a complex variable defined wherever this series converges.}$$

Indeed, as this transformation is an infinite series, it will only exist for the values of z for which this series converges.

Examples :

$$- x(n) = \delta(n) \Rightarrow X(z) = 1,$$

$$- x(n) = \delta(nk) \Rightarrow X(z) = z^{-k}, \quad \text{if } k > 0 \quad \text{ROC} = \mathbb{C} - \{0\} \quad \text{if } k < 0 \quad \text{ROC} = \mathbb{C} - \{\infty\}$$

$$- x(n) = (\underline{1}, 2, 3, 5, 0, 2) \text{ we can write } x(n) = \delta(n) + 2.\delta(n-1) + 3.\delta(n-2) + 5.\delta(n-3) + 2.\delta(n-5)$$

$$X(z) = 1 + 2z^{-1} + 3z^{-2} + 5z^{-3} + 2z^{-5} \quad \text{ROC} = \mathbb{C} - \{0\}$$

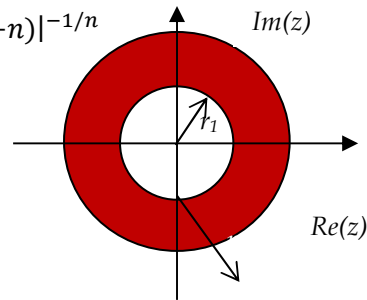
The set of values of the complex variable z for which the series converges is called Convergence Region (ROC):

$$RDC = \left\{ z \in \mathbb{C} / \sum_{n=-\infty}^{+\infty} |x(n).z^{-n}| < +\infty \right\}$$

In general, we show that the ROC is a convergence ring centered on the origin defined by:

$$r_1 < |z| < r_2 \quad \text{with } r_1 = \lim_{n \rightarrow +\infty} |x(n)|^{1/n} \text{ and } r_2 = \lim_{n \rightarrow +\infty} |x(-n)|^{-1/n}$$

where r_1 can be reduced to 0 and r_2 can be equal to ∞ .

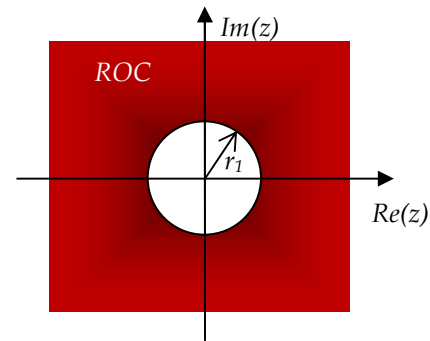
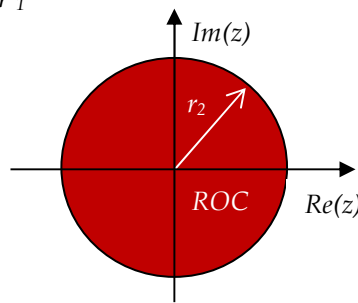


- $x(n)=0$ for $n < n_0 \Rightarrow r_2 = +\infty$,

ROC = region outside the circle of radius r_1

- $x(n)=0$ for $n > n_0 \Rightarrow r_1 = 0$

ROC = disk of radius r_2



\Rightarrow anti-causal system: ROC circle. causal system: ROC outside the circle.

Examples

- Let $a > 0$, $x(n) = \begin{cases} a^n & \text{si } n \geq 0 \\ 0 & \text{si non} \end{cases} \Rightarrow X(z) = \sum_{n=0}^{+\infty} a^n z^{-n} = \frac{z}{z-a}$, convergent for $|z| > a$.

- Let $b > 0$, $y(n) = \begin{cases} 0 & \text{si } n \geq 0 \\ -b^n & \text{si } n < 0 \end{cases} \Rightarrow Y(z) = \sum_{n<0} -b^n z^{-n} = \frac{z}{z-b}$, convergent for $|z| < b$.

- Let $a > 0$, $b > 0$, $w(n) = \begin{cases} a^n & \text{si } n \geq 0 \\ b^n & \text{si } n < 0 \end{cases} \Rightarrow W(z) = \frac{z}{z-a} - \frac{z}{z-b}$, convergent for $b > |z| > a$.

Note: The ZT of a^n for $n \in]-\infty, +\infty[$ does not exist.

The properties that are most used are summarized as follows:

If we define: $x(n) \xrightarrow{TZ} X(z)$, $x_1(n) \xrightarrow{TZ} X_1(z)$ and $x_2(n) \xrightarrow{TZ} X_2(z)$

- Linearity: $a \cdot x_1(n) + b \cdot x_2(n) \xrightarrow{TZ} a \cdot X_1(z) + b \cdot X_2(z)$ - Convolution: $x_1(n) * x_2(n) \xrightarrow{TZ} X_1(z) \cdot X_2(z)$

- Delay theorem: $x(n-k) \xrightarrow{TZ} z^{-k} \cdot X(z)$ - Advance: $x(n+k) \xrightarrow{TZ} z^k \cdot X(z)$ - $\sum_{n=0}^{k-1} x(n) z^{k-n}$

Examples

$$\Rightarrow X(z) = \frac{1}{2} \left(\frac{1}{1 - e^{j\omega_0} z^{-1}} + \frac{1}{1 - e^{-j\omega_0} z^{-1}} \right) = \frac{1 - \cos(\omega_0) z^{-1}}{1 - 2 \cos(\omega_0) z^{-1} + z^{-2}} \quad \text{with} \quad |z| > 1$$

2. Rational ZTs (corresponding to LTIs)

The invariant linear systems described by a finite difference equation have a rational Z transform, which is how these will be written as the ratio of two polynomials in z^{-1} .

$$\sum_{i=0}^M a_i y(n-i) = \sum_{i=0}^N b_i x(n-i) \xrightarrow{TZ} \sum_{i=0}^M a_i \cdot z^{-i} Y(z) = \sum_{i=0}^N b_i \cdot z^{-i} X(z)$$

An LI system can be characterized by $h(n)$ or by the Z transform ($H(z)$) of its impulse response $h(n)$, also called the transfer function of the system.

$$\Rightarrow H(z) = \frac{\sum_{i=0}^N b_i z^{-i}}{\sum_{i=0}^M a_i z^{-i}} = \frac{b_0}{a_0} z^{M-N} \frac{N(z)}{D(z)} = \frac{b_0}{a_0} z^{M-N} \frac{\prod_{i=1}^N (z-z_i)}{\prod_{i=1}^M (z-p_i)} = K z^{M-N} \frac{\prod_{i=1}^N (z-z_i)}{\prod_{i=1}^M (z-p_i)}$$

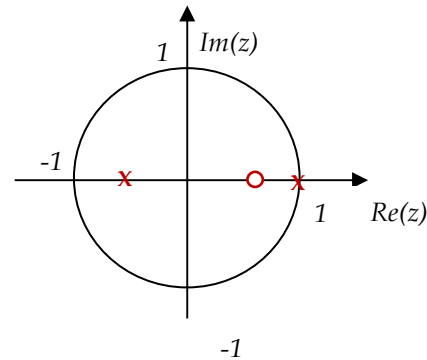
We call zeros the values of z for which $H(z)=0$ and we call poles the values of z for which $H(z)$ is infinite (cancels the denominator). Thus $H(z)$ has N zeros (z_i), M poles (p_i). If $M>N$, it has (MN) zeros at 0, otherwise (NM) poles at 0.

Thus, the position of its poles and its zeros (+ the amplitude factor $K = b_0 / a_0$) will provide us with a complete description of $H(z)$ (therefore of $h(n)$ and $H(f)$) therefore of the behavior of the system. $H(z)$ can therefore be represented in the form of a circle modeling the position of the poles and the zeros in the complex plane.

Example

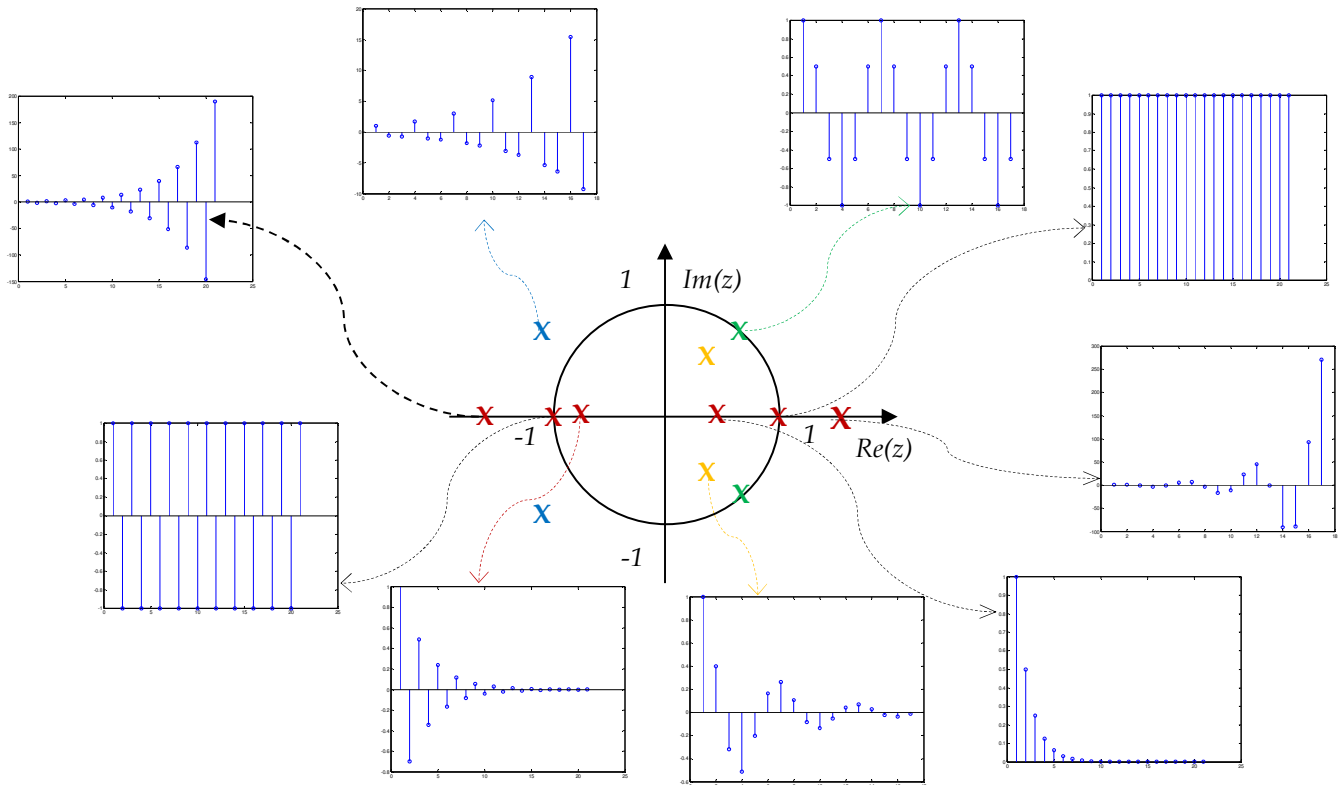
$$H(z) = \frac{3z-2}{(z-1)(z+0.5)}$$

A zero in $2/3$ and two poles $p_1 = -0.5$ and $p_2 = 1$



Remarks

- In most systems, the a_i and the b_i are real \Rightarrow the poles and the zeros are either real or complex conjugate pairs.
- Recall that the ROC of a causal system is the outside a circle. Moreover, if it is stable: $\sum_n |h(n)| < \infty$, since $H(z) = \sum_{n=-\infty}^{+\infty} h(n) \cdot z^{-n}$, so it suffices that $z=1$ be part of the ROC.
- For a causal and stable system, all the poles are inside the unit circle ($|p_i| < 1, \forall i$). The convergence domain cannot contain poles since the ZT does not converge at the poles. If it is anti-causal, it will be stable if the poles are outside the unit circle.
- If the filter is non-recursive $H(z) = \sum_{i=0}^N b_i \cdot z^{-i}$. An FIR filter has all its poles at the origin and will therefore always be stable.



- A simple or multiple pole p_i will correspond to an impulse response which converges if $|p_i| < 1$. It will diverge otherwise, i.e. if $|p_i| > 1$.
- Knowing that each complex pole is associated with a conjugate pole, this will give an oscillating impulse response $h(n)$ (cosine or sine) damped if $|p_{i=1,2}| < 1$ or divergent if $|p_{i=1,2}| > 1$.
- In a minimum phase system, all zeros are inside the unit circle ($|z_i| < 1, \forall i$).

- **Determination of the frequency response of digital filters**

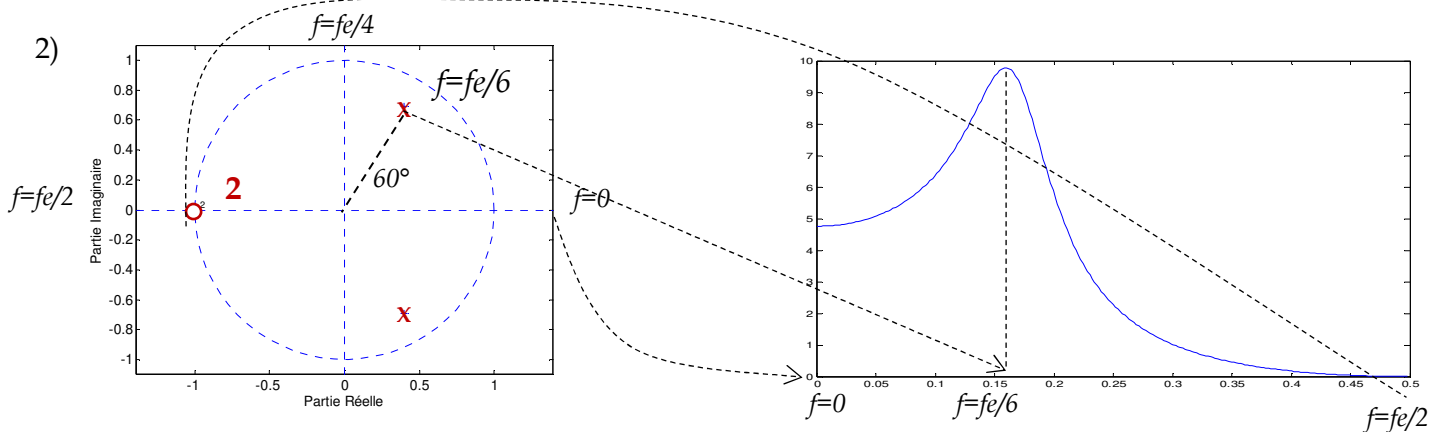
Assume that the unit circle ($|z| = 1$) \in ROC of $X(z)$. We restrict the calculation of $X(z)$ to the unit circle by setting $z = e^{2\pi j f T_e}$.

When a zero is placed on a given point of the z -plane, the frequency response will be 0 at the point considered. A pole on the other hand will produce a peak at the corresponding point. The closer the poles or zeros are to the unit circle, the more they influence the frequency response [11].

- a zero or a pole at the origin does not influence the modulus of the frequency response.
- a zero on the unit circle introduces a cancellation of the module for the corresponding frequency
- A zero near the unit circle introduces attenuation in the frequency response module. The greater the attenuation the closer the zero is to the unit circle.
- A pole on the unit circle introduces infinite resonance into the frequency response modulus for the corresponding frequency.

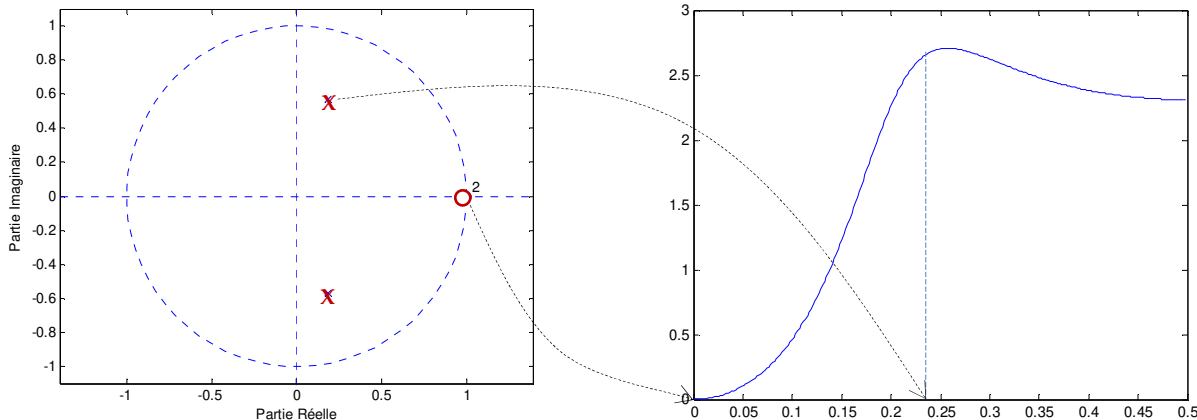
- A pole in the vicinity of the unit circle introduces a resonance all the more important in the module of the frequency response as the pole is close to the unit circle.

1) In the figure below the complete circle corresponds to a sampling frequency f_e . Poles near the unit circle cause large peaks while zeros near or on the unit circle produce minima. This plot will allow us to identify the nature of the filter. We can also have an idea of its general behavior: low pass, high pass or band pass, know its cutoff frequency(ies).



$$3) H(z) = \frac{z^2 - 2z + 1}{z^2 - 0.371z + 0.36} \Rightarrow \text{Double zeros in } z = -1, \text{ poles } p_{1,2} = \pm 0.6e^{j72^\circ}$$

- A double zero in $z = 1 \Rightarrow |H(f)| = 0$ for $f = 0$ - Poles close to the unit circle \Rightarrow maxima.



Note : Since the filter coefficients are real, the poles and zeros are real (on the real axis) or complex conjugate pairs.

- **Determination of the impulse response of FN (inverse ZT)**

The Z-transform has the advantage of being more easily invertible than the Fourier transform. The transition from ZT to $h(n)$ can be done through the Z transform of known elementary signals provided that it is possible to write $H(z)$ as the combination of elementary transforms. In the contrary case, one can employ integration on a closed contour by using the calculus of the residues, or the development in power of z and of z^{-1} or the development in elementary fractions [12].

1. The general relation of the inverse z -transform is given by the equation given by the Cauchy integral: $x(n) = \frac{1}{2\pi j} \oint_C X(z) \cdot z^{n-1} \cdot dz$, where C is a closed contour traveled counter-clockwise containing the origin.

In practice, we use the residue theorem: $x(n) = \sum_{p_i \text{ poles de } z^{n-1} X(z)} \text{Res} [z^{n-1} X(z)]_{z=p_i}$

$$\text{Res} [z^{n-1} X(z)]_{z=p_i} = \frac{1}{(m-1)!} \frac{d^{m-1}}{dz^{m-1}} [(z-p_i)^m z^{n-1} X(z)]_{z=p_i}$$

2. Inverse transform by polynomial division: It is possible to calculate the inverse Z transform according to the increasing powers of z^{-1} (causal system) or according to the decreasing powers of z (anti-causal system).

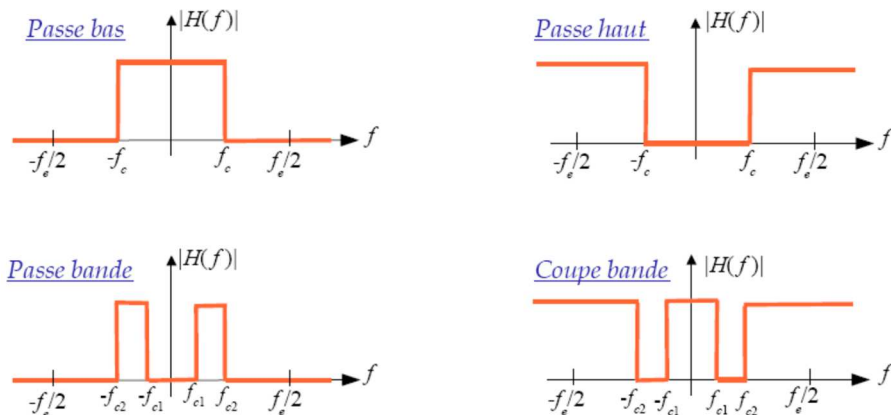
$$X(z) = \sum_n C_n z^{-n} \xrightarrow{TZ^{-1}} x(n) = C_n$$

3. The general idea of this approach consists of finding for a complex function $X(z)$ an expansion into simpler Z functions for which an inverse transform is known: $X(z) = \sum_i X_i(z) \xrightarrow{TZ^{-1}} x(n) = \sum_i x_i(n)$

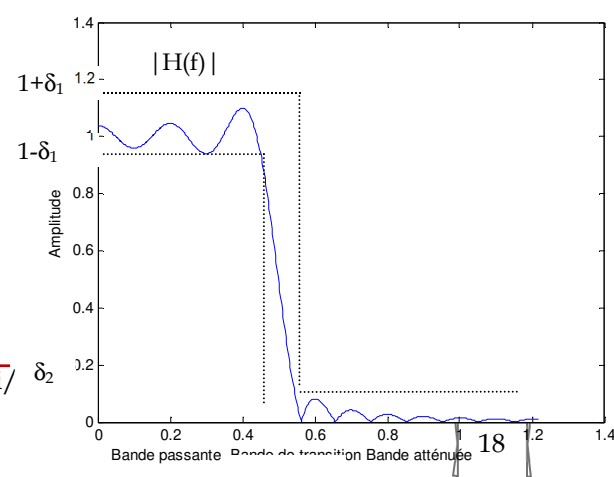
where the $X_i(z)$ are functions whose ZT^{-1} are known (See page 38).

3. Characteristics of digital filters

A digital filter consists of a group of logic circuits subject to a calculation process (or algorithm) which gives this filter a determined function (low-pass, high-pass, band-pass, band rejector, integrator [$y(n) = (x(n) + x(n-1))/2$], differentiator [$y(n) = (x(n) - x(n-1))/2$], ...). It must be pointed out that some filters are not designed to stop a frequency, but to modify the gain slightly at different frequencies, such as equalizers. They are all linear, discrete, time-invariant, one-dimensional systems. Moreover, for them to be physically realizable, they must necessarily be causal.



The filters pictured above are ideal. In a real case it is not possible to obtain such a steep cutoff frequency. The passage between pass-through zones and attenuated zones is done by so-called "transition" zones f_p - f_a whose width will express the selectivity of the filter. Moreover, the pass and attenuate bands are also not ideal, they contain ripples whose amplitude is expressed by the ripple parameters in passband δ_1 and attenuated band δ_2 [9].



- **Linear phase filter:**

Ideally, it is desirable for a filter to have a linear phase in the passband. A linear phase will ensure the same phase shift for all frequencies (no distortion). FIR filters can generate linear phase filters provided that the impulse response is symmetrical.

If a filter is linear phase, its frequency response is of the form:

$$H(f) = R(f)e^{-j\phi(f)} \text{ with } \phi(f) = \phi_0 + 2\pi f\tau$$

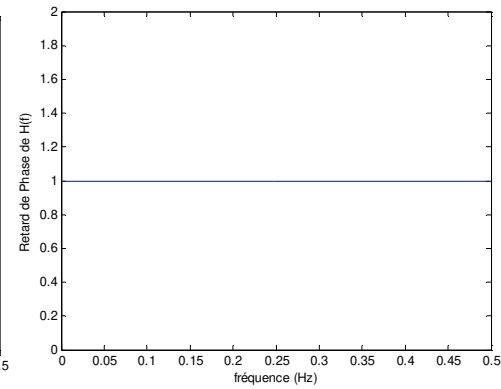
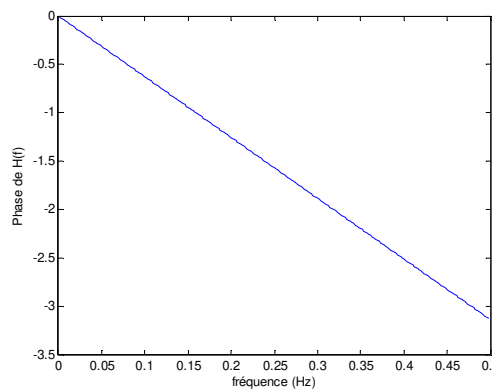
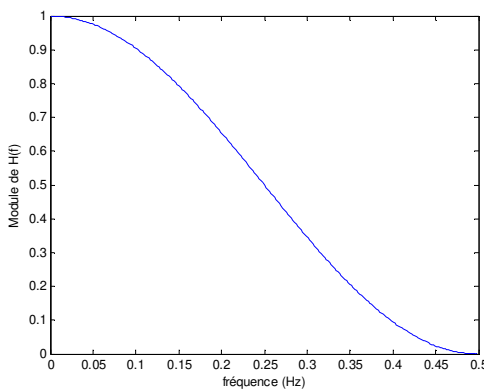
And the derivative of the latter with respect to f provides the 'group delay', defined therefore by:

$$\beta = -\frac{d\phi(f)}{df}$$

and which corresponds to the delay undergone by the signal after passing through a filter. If the phase is linear (symmetrical FIR filters), the delay is constant and the signal at the output will therefore have minimal distortion since the effect of the phase on the signal will be a simple time shift (essential in an audio system)

Example: $y(n) = \frac{1}{4}(x(n) + 2x(n-1) + x(n-2))$

$$\Rightarrow h(n) = \frac{1}{4}(\delta(n) + 2\delta(n-1) + \delta(n-2)) \text{ And } H(f) = e^{-2\pi jf} \cos^2(\pi f)$$



- **Minimum phase filter:** For an IIR filter, the group delay is not constant. We want to have a minimum of delay by orienting ourselves to make minimum phase filters. A filter is said to be minimum phase if it and its inverse are both stable and causal. This implies that all the poles and zeros of the transfer function $H(z)$ are inside the unit disk. A minimum phase filter has an impulse response (coefficients of $H(z)$) with a decreasing tendency because its zeros are in modulus less than or equal to unity.

Example: $y(n) = 0.75x(n) + 0.25x(n-1)$

$y(n)$ is obtained rather from "recent" samples of the input signal than from "old" samples. Note that any causal and stable filter can be decomposed into a product of minimum-phase filters by an all-pass cell.

- **IIR or FIR ?**

➤ IIR filters have the advantage that they are effective. With very few poles and zeros we can provide most of the frequency responses that we may need in audio applications. However, because the filter is recursive, errors in numerical precision become a matter of importance, as they can amplify and get out of

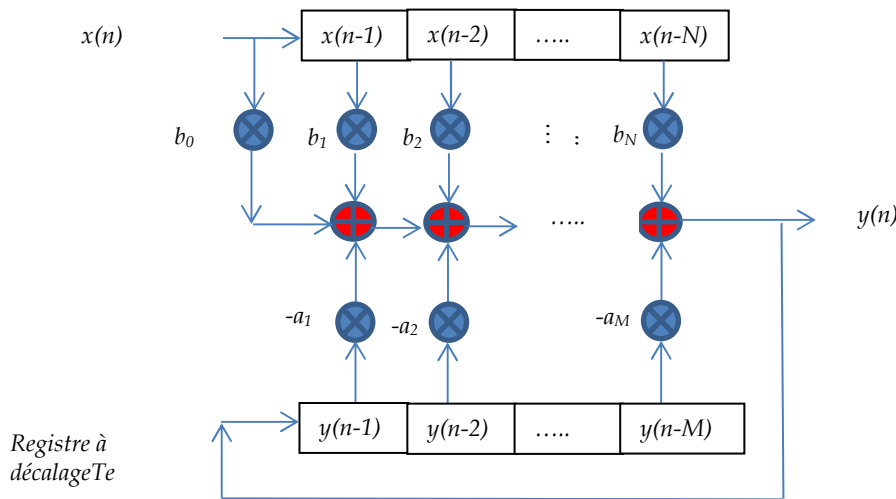
control, first in the form of noise, but eventually in the form of jitter. The shape of the impulse response is not easy to determine, either, because it is defined indirectly by the poles and zeros of $H(f)$.

➤ On the other hand, FIR filters never have instability problems, because the output is only a finite sum of samples of the input. However, when the impulse response is long, the number of operations can become a deciding factor when choosing between FIR or IIR. Another advantage of FIRs is the constant group delay, which allows for minimal phase distortion in the processed signal [13].

- **Structure of digital filters**

Applying a digital filter involves calculating the output $y(n)$ at time $t=nT_e$ from the previous outputs and inputs plus the current value of the input.

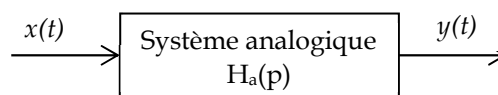
$\sum_{i=0}^M a_i y(n-i) = \sum_{i=0}^N b_i x(n-i)$ Taking $a_0=1$, we get

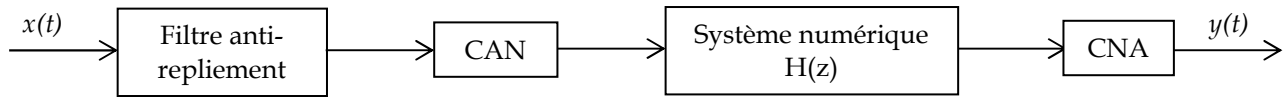


A digital filter generally consists of the following elements: one or more delay elements (these are shift registers playing the role of delayed memories), controlled by a period clock; arithmetic operators (adders and multipliers); registers providing the weighting coefficients of the filter [18]. Thus, at each clock tick T_e , the values of the registers undergo an offset making it possible to calculate the new output. This structure can also be realized by software.

4. Synthesis of Digital Filters and Analog Template

The synthesis of a filter is a set of processes which begins with the definition of the characteristics of the filter, up to its computer and/or electronic production, passing through the determination of its coefficients. To synthesize a digital filter, the template of the analog filter is considered to be known and a digital system is sought characterized by a transfer function $H(z)$ to be inserted in the circuit above making it possible to satisfy the analog template.





The determination of the transfer function of a digital filter, by a direct method, is not always very simple. On the other hand, the problem which consists in transforming an analog filter into a digital filter is relatively simple. Therefore, many methods are proposed to design a digital filter from the equivalent analog filter. In any case, the synthesis of a digital filter is an approximation of an equivalent ideal analog filter. It is necessary to constrain a certain number of parameters.

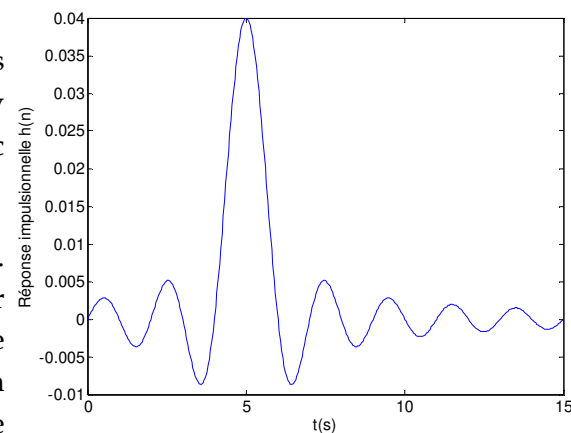
The synthesis of a digital filter includes the following steps:

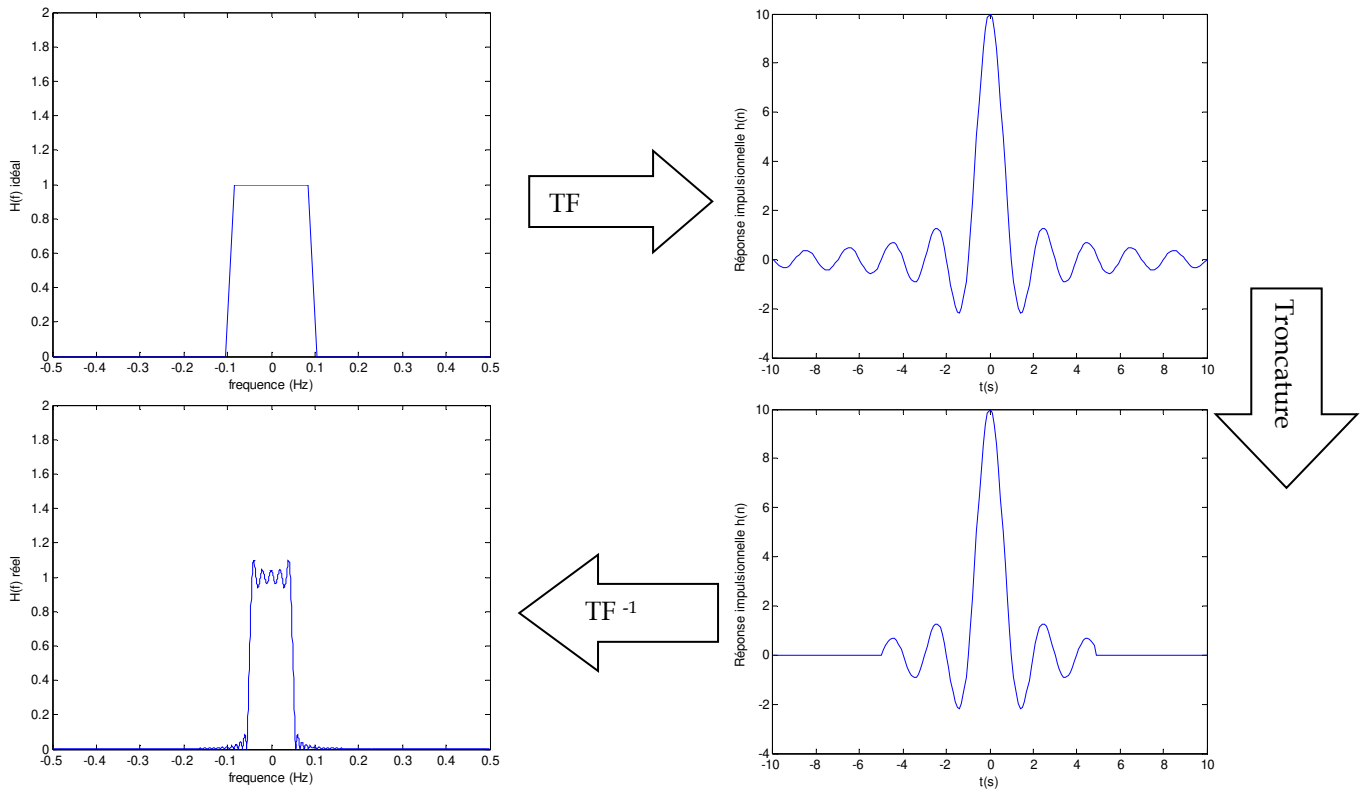
1. determining a desired ideal frequency response;
2. the determination of the best approximation under a certain number of constrained constraints (stability, speed, precision, linear phase shift, etc.);
3. the choice of a calculation structure realizing the approximated filter.

The model functions used for filter synthesis are either the impulse response or the frequency response (the latter is preferred) of known analog filters. If one employs the impulse response, the elements $h(n)$ of the digital impulse response are obtained by calculating $h(t)$, the impulse response of the analog filter, at times $t=nT_e$.

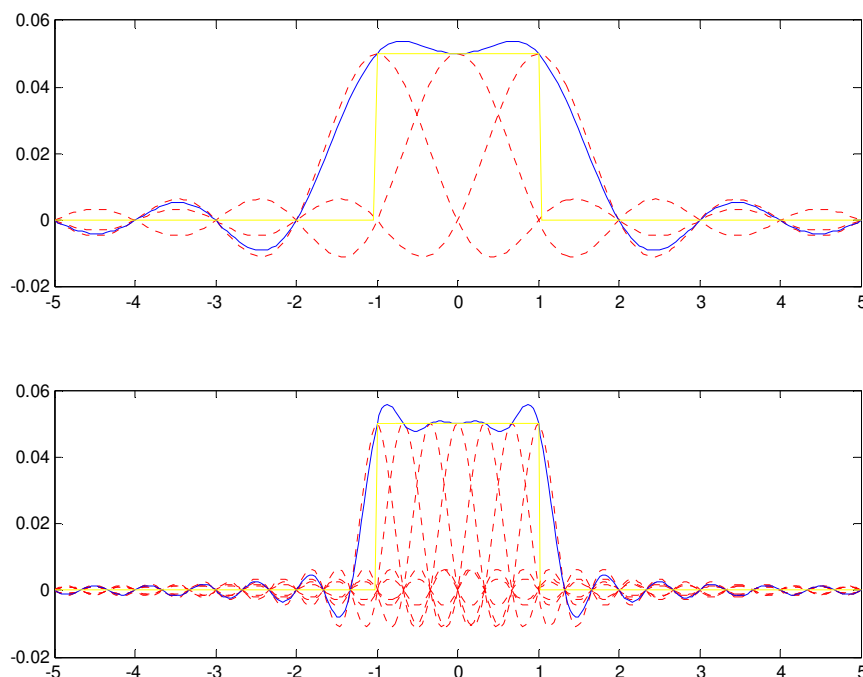
Recall that the ideal filters have a linear phase shift and are not physically feasible, because the ideal frequency responses correspond to a non-causal temporal response. For example, by considering the low-pass filter $H(f) = \Pi(f) e^{-2\pi j f T}$, we will obtain a sinc shifted by T .

One can observe that it is necessary to cancel part of the signal. As a result, it is no longer possible to obtain an ideal low-pass filter (straight and with a perpendicular transition line). It follows that the filters which will be able to be really synthesized do not have a frequency response corresponding to the gate function, but will be able to closer.





As shown in the following figure, truncation (multiplication by a gate of width $N T_e$) of the sinc in the time domain will result in a convolution in the frequency domain of the ideal filter with a sinc vanishing every $N T_e$. For large values of N , the sincs in the passband will compensate each other but around the points of discontinuity (cutoff frequency), the ripples remain apparent.



It can be observed that the differences with respect to the ideal filter (ie the gate function) are mainly the ripples in the passband and in the attenuated band as well as the width of the transition.

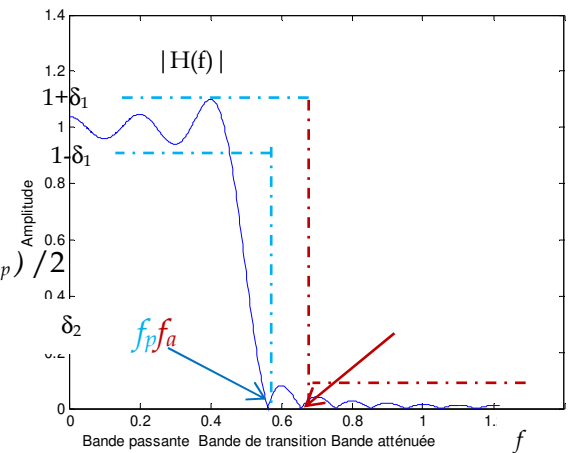
This is how the filter specifications will be defined by a linear frequency template or in dB (decibels). This template indicates the cutoff frequency or frequencies, the width of the desired minimum transition band,

the maximum ripple of the passband and of the attenuated band, the sampling frequency and possibly the maximum allowed order.

Filter template

The template of a filter is none other than the set of characteristics of the filter, namely:

- the bandwidth (BP) from 0 up to f_p
- the attenuated band (or cut BA) from f_a to $f_e/2$
- The gain of the filter in the passband.
- The attenuation of the notched band filter f_a .
- the width $\Delta f = f_a - f_p$ of the transition zone $\Rightarrow f_c = f_a + \Delta f / 2 = (f_a + f_p) / 2$
- the amplitude of the oscillations in bandwidth:
 $\delta_1 \Rightarrow A_p = 20 \log(1 + \delta_1)$ ripple allowed in BP
- the amplitude of the ripples in the attenuated band:
 $\delta_2 \Rightarrow A_a = -20 \log(\delta_2)$ ripple allowed in BA



In practice, the closer the frequencies f_a and f_p are, the higher the order of the filter must be. For an ideal filter, these values would be confused.

5. Synthesis of FIR filters by the window method

The use of FIR filters can turn out to be attractive in view of its many advantages: unconditional stability (all the poles are at 0), possible linear phase. However, they have the disadvantage of requiring a greater number of coefficients than IIR filters to obtain the same frequency characteristics due to the absence of poles outside 0. Thus, any stable and causal digital filtering function can be approximated by the transfer function of an FIR filter.

Recall that the output of an FIR filter will be expressed as a linear combination of a finite set of input elements:

$$y(n) = \sum_{i=0}^{N-1} b_i x(n-i) \quad \text{from where} \quad H(z) = \sum_{n=0}^{N-1} b_n \cdot z^{-n} \Rightarrow H(f) = \sum_{n=0}^{N-1} b_n \cdot e^{-2\pi j f n T_e}$$

Thus, the weighting coefficients are nothing but the values of the impulse response of the filter. These coefficients constitute the coefficients of the Fourier series development of the transfer function $H(f)$ (see DTFT chapter 1)

Because an FIR filter has a polynomial (non-rational) transfer function, it cannot be obtained by transposing a continuous filter. The two most used methods for approximating digital filters FIR are SO:

- Development by Fourier series: this series is then truncated by window functions to limit the impulse response. The Fourier coefficients coincide with the samples of the Filter's impulse response.
- Frequency Response Sampling: This method uses DFT. This is applied to the desired coefficients b_i to obtain a frequency sequence which corresponds to the frequency response of the filter.

There are other methods such as optimization methods which are based on the minimization of an error criterion between the real curve and the ideal filter [9].

The window method consists, knowing the analytical expression $H(f)$ of the continuous frequency response (whose known mathematical formulation) approach, to be determined by use of the inverse discrete-time FGE, USTHB [assiakourgli@gmail.com / <http://perso.usthb.dz/~akourgli/>]

Fourier transform, the response impulsive. This obtained non-causal temporal response will be delayed to make it causal [18]. So :

1. From the ideal template of the filter, the coefficients of the filter are determined by limiting the calculation to N values distributed symmetrically around $n=0$. Then, we calculate the inverse DTFT of the ideal filter which will allow us to find the samples of the impulse response are the coefficients of the filter:

$$h(n) = \begin{cases} \frac{1}{f_e} \int_{-f_e/2}^{f_e/2} H(f) e^{2\pi j f n T_e} df & \text{Nimpair (Filtredetype } \rightarrow I) \\ \frac{1}{f_e} \int_{-f_e/2}^{f_e/2} H(f) e^{-j\pi f T_e} e^{2\pi j f n T_e} df & \text{Npair (Filtredetype } \rightarrow II) \end{cases}$$

Notes :

- The type I filter is the most used (it is the one we develop in what follows) except for the differentiators.
- The type II filter has a zero in -1 (Number of zeros is odd therefore distribution of zeros in conjugate pairs + a zero in $f_e/2$ since a sum of cosines), it cannot therefore be used for a high pass or a tape cutter.
- There are filters of type III (N odd) and IV (N even) making it possible to obtain an anti-symmetrical impulse response with also linear phase shift (sum of sines instead of cosines). They both have zeros at 1 (sin at $f=0$ is zero), so their use is not suitable for low-pass filters. In addition, the type III filter also has a 0 in -1, it can only be used for a band pass.

2. This method produces an infinite series of coefficients, we then limit the impulse response to N samples (truncation). Knowing that the truncation induces ripples, we can use the weighting windows to attenuate them. Thus, the ideal impulse response $h(n)$ will be multiplied by the discrete window $w_N(n)$ of length N:

$$h'_N(n) = h(n) \cdot w_N(n). \text{ (Choice of window: see chapter 1)}$$

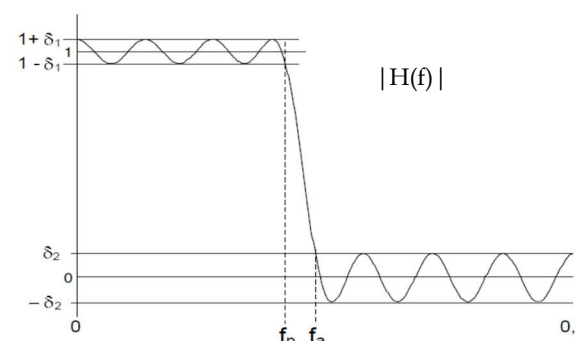
3. It only remains to shift the impulse response $h'(n)$ to have a causal solution.

Note: For the choice of f_c it will be necessary to do pay attention to the cutoff frequencies to be taken into account. In order to have good results during the synthesis, it is not the cutoff frequencies of the ideal filter that must be used, but it is necessary move them to center them in the transition area. For a low pass, the increase of the half transition zone (Δf), i.e. $f_p + \Delta f$ and for a high pass, decrease it by the half transition zone. For a band pass, decrease the first cutoff frequency of the half transition zone and increase the second of the half transition zone, for a band rejector, we will do the opposite.

Example :

- Calculation of the ideal impulse response (case N odd)

$$h(n) = \frac{1}{f_e} \int_{-f_e/2}^{f_e/2} H(f) e^{2\pi j f n T_e} df$$



$$\text{We set } f_c = (f_p + f_a)/2 \Rightarrow h(n) = \frac{1}{f_e} \int_{-f_c}^{f_c} e^{2\pi j f n T_e} df = \frac{1}{2\pi j n f_e T_e} e^{2\pi j f n T_e} \Big|_{-f_c}^{f_c}$$

$$\Rightarrow h(n) = \frac{e^{2\pi j f_c n T_e} - e^{-2\pi j f_c n T_e}}{2\pi j n} = \frac{\sin(2\pi f_c n T_e)}{\pi n} = \frac{\sin(\pi n f_c / (f_e/2))}{\pi n} \quad f$$

We normalize the frequencies with respect to $f_e/2$ (or f_e) i.e. we replace everywhere f_c by $f_c/(f_e/2)$, we then obtain :

$$h(n) = \frac{\sin(\pi n f_c / (f_e/2))}{\pi n} \frac{f_c / (f_e/2)}{f_c / (f_e/2)} = f_c \frac{\sin(\pi n f_c)}{\pi n f_c}$$

Just as easily, one can determine the impulse responses of a given highpass ($1-H(f)$), bandpass (difference of 2 lowpasses), and bandstop.

The values of the ideal impulse response $h(n)$ are given in the following table. The frequencies f_c indicated in this table (desired cut-off frequencies) are expressed also in normalized frequencies (divided by $f_e/2$).

	Impulse response $h(n)$	
Filter type	$h(n)$ for $n \neq 0$	$h(n)$ $n=0$
low pass	$f_c \frac{\sin(\pi n f_c)}{\pi n f_c}$	f_c
high pass	$-f_c \frac{\sin(\pi n f_c)}{\pi n f_c}$	$1-f_c$
band pass	$f_{c2} \frac{\sin(\pi n f_{c2})}{\pi n f_{c2}} - f_{c1} \frac{\sin(\pi n f_{c1})}{\pi n f_{c1}}$	$f_{c2} - f_{c1}$
band rejector	$f_{c1} \frac{\sin(\pi n f_{c1})}{\pi n f_{c1}} - f_{c2} \frac{\sin(\pi n f_{c2})}{\pi n f_{c2}}$	$1-(f_{c2} - f_{c1})$

- Limitation of the number of samples to N

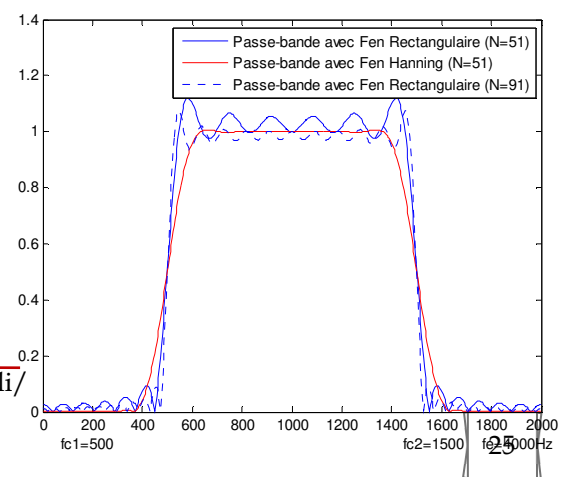
$$h'_N(n) = h(n) \cdot w(n)$$

$$\text{if } w(n) = \begin{cases} 1 & |n| \leq \frac{N-1}{2} \\ 0 & \text{ailleurs} \end{cases} \Rightarrow |W(f)| = \frac{\sin(N\pi f)}{\sin(\pi f)} H'_N(f) = H(f) * W(f)$$

Temporal truncation introduces ripples and induces a slower transition zone determined by the width of the main lobe. A compromise has to be made between the stiffness and the amplitude of the undulations. Note that this method gives ripples of the same amplitude in the pass band and in the attenuated band.

Note: If N increases, the extent of the oscillations decreases (the response is flatter) and the width of the transition band decreases ($\Delta \text{low } f$). It can be observed that the strongest oscillations tend to concentrate at the discontinuities. The latter do not decrease if N increases: there is always an overshoot roughly equal to 9% concentrating at the points of discontinuity: this is the Gibbs phenomenon (See LW n°1).

FGE, USTHB [assiakourgli@gmail.com / <http://perso.usthb.dz/~akourgli/>



To reduce the oscillations: the weighting windows are used which make it possible to obtain strong attenuation of the oscillations but this is done to the detriment of the width of the transition band which becomes greater.

For the choice of the weighting window, one will proceed as follows: Depending on the attenuation δ_2 required in the specification of the filter, one will choose the type of window $w(n)$ to be used. Then, depending on the width of the transition zone Δf (also specified at the start) and the type of window $w(n)$, the length of the impulse response N will be determined.

Windows $w_N(n)$	Width of Transition: $\Delta f (2 \Delta f / f)$	Attenuation in attenuated band A
$w_{Rect}(n) = \begin{cases} 1 & \text{pour } n \leq \frac{N-1}{2} \\ 0 & \text{ailleurs} \end{cases}$	$1.8/N$	21
$w_{Ham}(n) = \begin{cases} 0.5 + 0.5 \cos(\frac{2\pi n}{N-1}) & \text{pour } n \leq \frac{N-1}{2} \\ 0 & \text{ailleurs} \end{cases}$	$6.2/N$	44
$w_{Ham}(n) = \begin{cases} 0.54 + 0.46 \cos(\frac{2\pi n}{N-1}) & \text{pour } n \leq \frac{N-1}{2} \\ 0 & \text{ailleurs} \end{cases}$	$6.6/N$	53
$w_{Black}(n) = \begin{cases} 0.42 + 0.5 \cos(\frac{2\pi n}{N-1}) + 0.08 \cos(\frac{4\pi n}{N-1}) & \text{pour } n \leq \frac{N-1}{2} \\ 0 & \text{ailleurs} \end{cases}$	$11/N$	74

Example : We want to synthesize a low-pass filter with a cutoff frequency $f_c = f_e / 10$ with $\Delta f = f_e / 5$ and an attenuated band ripple > 50 db (see Laboratory Work n°3)

We normalize the frequencies $f_c / (f_e / 2) \Rightarrow f_c = 0.2 \Delta f / (f_e / 2) \Rightarrow \Delta f = 0.4$

a- $h(n) = 0.2 \frac{\sin(\pi n/5)}{\pi n/5}$

b- We choose $w(n)$ as the Hamming window

($A_a = 20 \log(\delta_2) = -53$) $\Rightarrow \Delta f = 6.6/N$

c- We calculate $N = 6.6 / \Delta f = 16.5$ we take $N = 17$

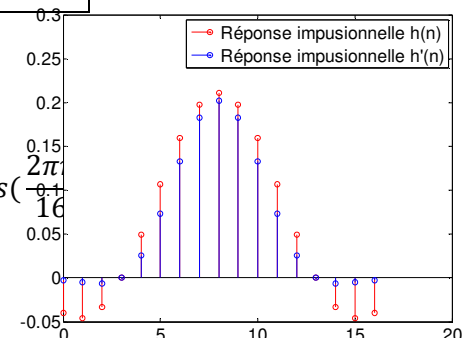
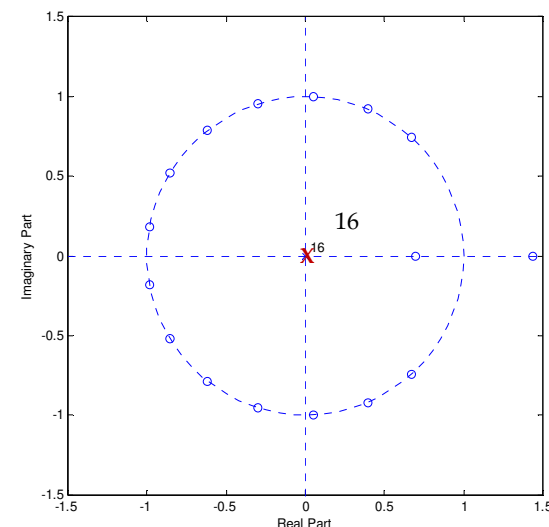
d- we calculate the values $h(n) = 0.2 \frac{\sin(\pi n/5)}{\pi n/5}$ for $-8 \leq n \leq 8$

NOT	-8	-7	-6	-5	-4	-3	-2	-1	0
$h(n)$	-0.0399	-0.0456	-0.0329	0	0.0493	0.1064	0.1597	0.1974	0.2110
NOT	1	2	3	4	5	6	7	8	
$h(n)$	0.1974	0.1597	0.1064	0.0493	0	-0.0329	-0.0456	-0.0399	

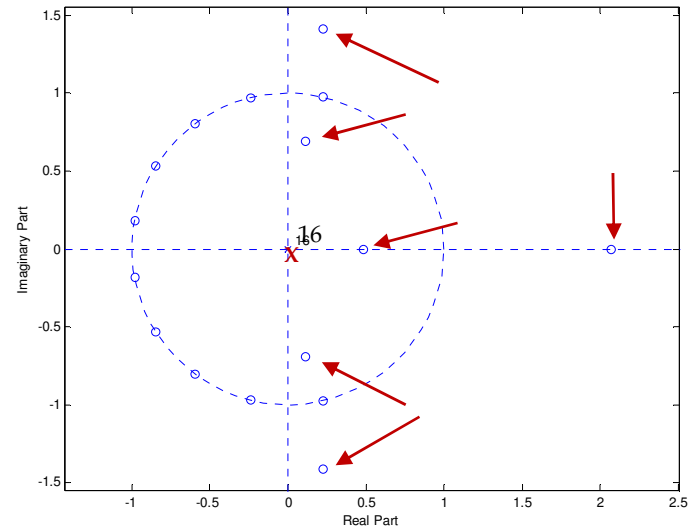
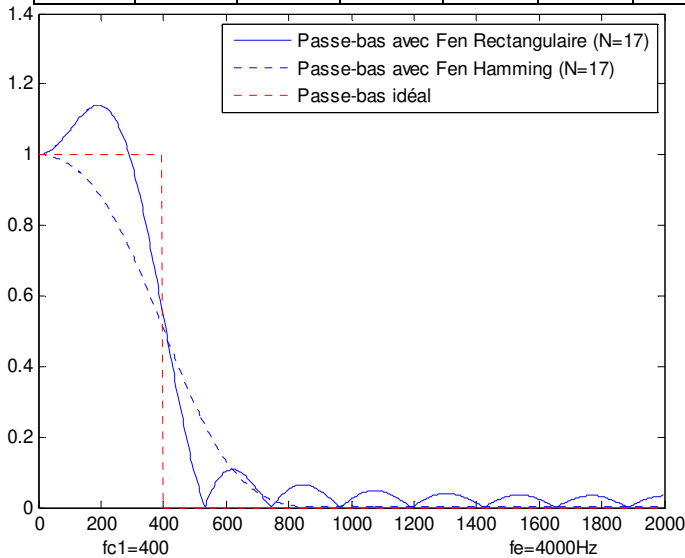
e- We multiply $h(n)$ by $w(n)$ to find $h'_N(n) = h(n) \cdot w(n)$

$$h'_N(n) = 0.2 \frac{\sin(\pi n/5)}{\pi n/5} \left[0.54 + 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \right]$$

f- We translate the result of 8 samples.



	0	1	2	3	4	5	6	7	8
$h(n)$	-0.0399	-0.0456	-0.0329	0	0.0493	0.1064	0.1597	0.1974	0.2110
$h'_N(n)$	-0.0031	-0.0050	-0.0067	0	0.0255	0.0730	0.1325	0.1826	0.2023
NOT	9	10	11	12	13	14	15	16	
$h(n)$	0.1974	0.1597	0.1064	0.0493	0	-0.0329	-0.0456	-0.0399	
$h'_N(n)$	0.1826	0.1325	0.0730	0.0255	0	-0.006	-0.005	-0.0030	



We can notice the location of zeros around the first 0 in 1 makes it possible to attenuate the secondary lobes and to maintain a cste answer around the zero.

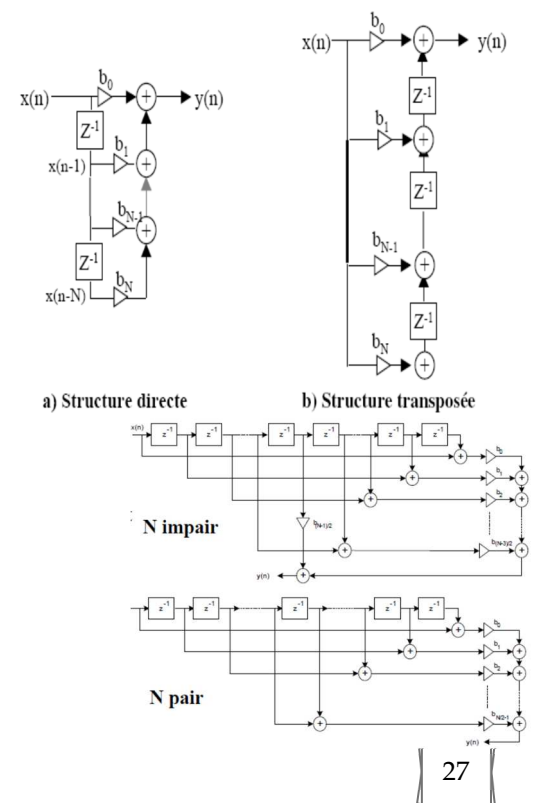
Constitution and realization of FIR digital filters

The concrete realization of a digital filter will in fact consist in materializing the calculation algorithm for the retained structure. We will have the possibility of working: Either in hard-wired logic (assembly of logic components, such as gates, memories, etc.), or in programmed logic (organization around a signal processing processor (DSP) or, same, use of a standard microprocessor (microcomputer).

The direct canonical structure (transversal or non-recursive) is given opposite:

An FIR filter requires $(N-1)$ multiplication operations, N addition operations for each new sample to be filtered.

One can also express the complexity as a multiplication-accumulation number which, in the case of FIR filtering, is N [9]. The memory cost of FIR filtering is $2(N+1)$ [($N+1$) coefficients b_i and ($N+1$) memory points for the vector of inputs $x(i)$]. If the sampling frequency of the input signal is equal to f_e , this means that the filter will have to be produced in a calculation time T less than $T_e = 1/f_e$. For a constant group delay filter, the structure will be as follows:



6. Synthesis of IIR recursive filters

The principle of defining the specifications of a recursive filter (IIR) takes place in the same way as for a non-recursive filter (FIR) but the synthesis will be done in a different way. The advantage of using an IIR filter lies mainly in the possibility of obtaining a narrow transition band for a reasonable order although it presents on the one hand, a risk of instability due to a high numerical sensitivity of the coefficients but which can however be controlled by determining a better adapted structure, and on the other hand, a highly non-linear phase variation.

Recall that for an IIR filter, the output is expressed as a linear combination of a finite set of input and output elements:

$$y(n) = \sum_{i=0}^N b_i x(n-i) - \sum_{i=1}^M a_i y(n-i) \quad \text{from where} \quad H(z) = \frac{\sum_{i=0}^N b_i \cdot z^{-i}}{1 + \sum_{i=1}^M a_i \cdot z^{-i}}$$

The direct method consists of placing poles and zeros at the useful frequencies, but the most common (indirect, also called transposition) is the use of analog filter synthesis methods resulting in a function $H(p)$ corresponding to the specifications. A function allowing the passage of the plane p in the plane z ($p = fct(z)$) is then used to obtain $H(z)$. This function must maintain the stability of the analog filter and maintain, at best, the characteristics of the frequency response $H(f)$ of the digital filter.

• Reminders on analog filters

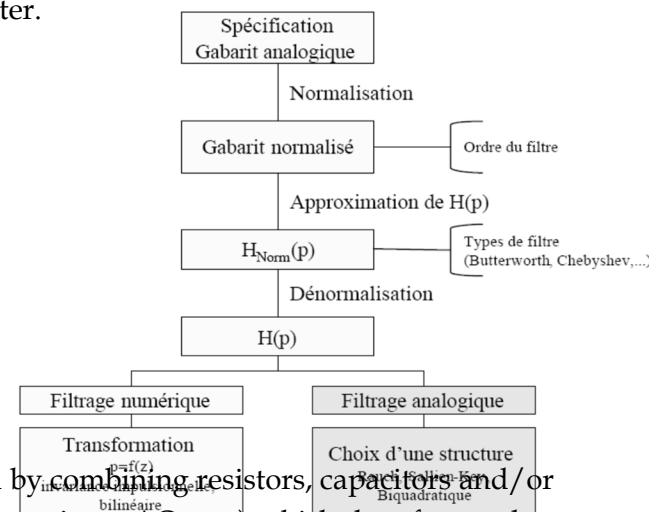
There are many methods to synthesize a recursive digital filter from an analog filter taken as a model [18]:

- the filter must have an imposed impulse or index response: these are the impulse invariance methods.
- the filter must have a frequency response falling within a given template: this is the bilinear transformation.

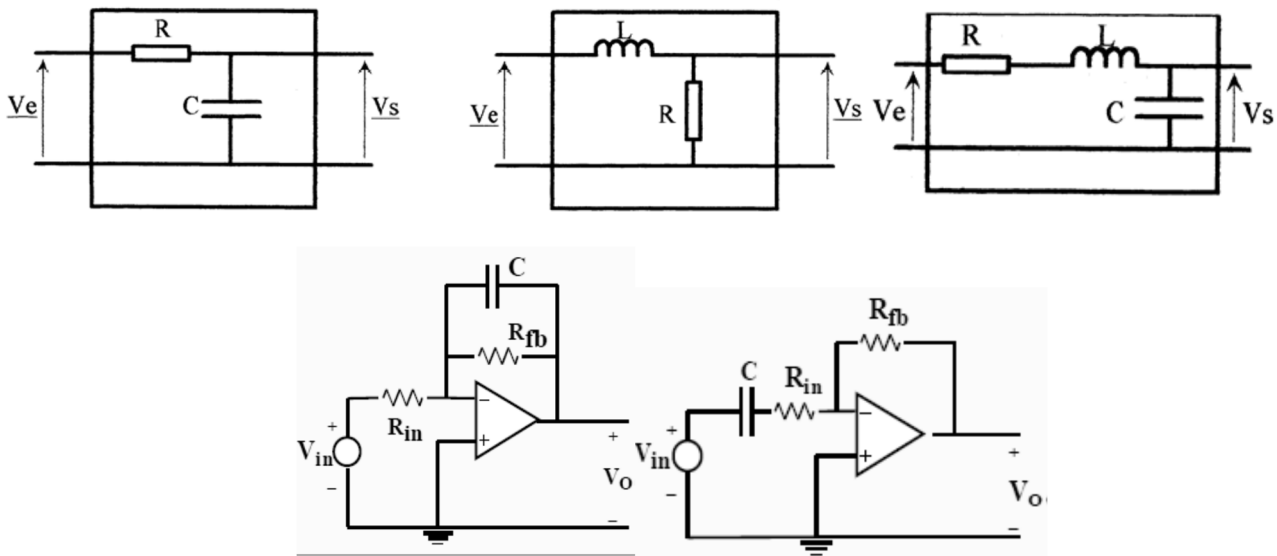
To design an analog filter, one can use passive filters obtained by combining resistors, capacitors and/or coils or even use active filters comprising an amplifier element (transistor, AO, etc.) which therefore makes it possible to modify the amplitudes signals. Thus, a basic (first order) passive filter can be composed of an RC cell or an RL cell.

It should be noted that:

- digital filters are limited to frequencies $< 100\text{MHz}$
- passive filters (L, C, quartz, etc;) are used for high frequencies
- active filters (R, C, ALI) use integrated linear amplifiers (ALI) limited to 1Mhz
- switched capacitance filters (integrated R and C, ALI, MOS controlled switch) have programmable frequencies ($< 10\text{ Mhz}$)



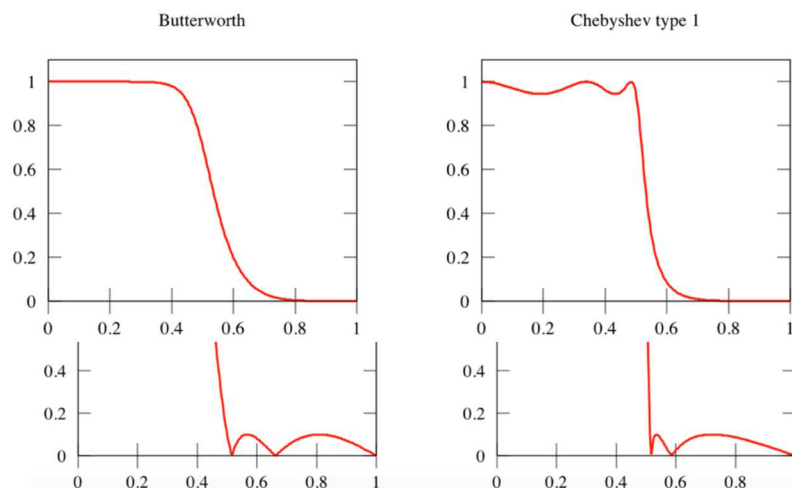
Below are some sample structures for first and second order low pass passive filters followed by first and second order low pass active filters as well.



To obtain filters of higher order N , one will employ filters of the first and the second order put in cascade. The asymptotic slopes will be proportional to the number of cells ($+N \uparrow$ and $+ \Delta f \downarrow$). Also note that low-pass or high-pass filters can have an integer order (1, 2, 3...) while band-pass or band-stop filters can only have an even order (2, 4, 6, ...) because they are made up of pairs of cells: 2 RC cells or an RC cell and an RL cell.

To synthesize analog filters meeting a given template, one will choose from a set of tested and proven analog filters therefore known for their properties in terms of attenuation slope and ripple in the passband and attenuated such as:

- Butterworth filters: Not very steep cutoff but constant bandwidth gain
- Chebyshev filters (Chebyshev): Significant cutoff stiffness but ripples in the passband (Chebyshev 1) or attenuated (Chebyshev 2) and simple to implement
- Cauer filters (also known as elliptical): Extremely steep cutoff but ripples in the passband and attenuated but more complex circuits to achieve.
- Bessel filters: Constant group delay but poor selectivity even for high order.



Note: The Bessel filter is useful for filtering broadband signals (high-speed HF modulations: PSK, 8-PSK, OFDM, etc.) while preserving the phases. It has no interest in the field of digital filtering.

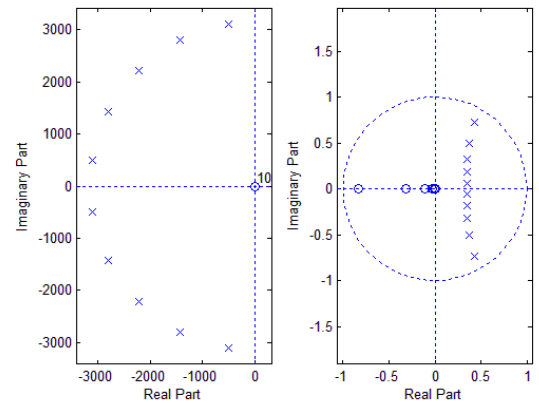
1. A Butterworth filter is characterized by the fact that the amplitude response is maximally flat in the passband and monotonically decreasing from a certain frequency (cutoff frequency).

The amplitude of an N-order Butterworth low-pass analog filter is defined by the expression:

$$|H(\omega)| = \frac{1}{\sqrt{1 + \left(\frac{\omega}{\omega_c}\right)^{2N}}} \quad \text{Or} \quad |H(\omega)| = \frac{1}{\sqrt{1 + \varepsilon^2 \left(\frac{\omega}{\omega_c}\right)^{2N}}}$$

ω_c is the cutoff pulsation limits and ε is a design parameter which sets the tolerance region in the passband $\delta_1 = 1/(1 + \varepsilon^2)^{1/2}$. The analog filter will have N poles placed on the unit circle

Ordre du filtre	Dénominateur (le numérateur est à 1)
2	$p^2 + \sqrt{2}p + 1$
3	$(p^2 + p + 1)(p + 1)$
4	$(p^2 + 1.8477p + 1)(p^2 + 0.7653p + 1)$
5	$(p^2 + 1.6180p + 1)(p^2 + 0.6180p + 1)(p + 1)$
6	$(p^2 + 1.9318p + 1)(p^2 + \sqrt{2}p + 1)(p^2 + 0.5176p + 1)$



The order N is determined by the transition zone ($\omega_a - \omega_p$) and the ripples allowed in bandwidth $A_p = 20 \log(1 + \delta_1)$ and attenuated band ($A_a = -20 \log \delta_2$)

$$N \geq \frac{\log\left(10^{\frac{A_a}{10}} - 1\right)}{2 \log(\omega_a / \omega_p)}$$

$$N \geq \frac{\log\left(10^{\frac{A_a}{10}} - 1\right) / \left(10^{\frac{A_p}{10}} - 1\right)}{2 \log(\omega_a / \omega_p)} = \frac{\log\left(\left(\frac{1}{\delta_2^2} - 1\right) / \left(\frac{1}{\delta_1^2} - 1\right)\right)}{2 \log(\omega_a / \omega_p)}$$

Note: The higher N, the greater the selectivity (Δ narrow f transition zone) but an IIR filter due to its recursion has an initialization period before the steady state which . Moreover, its phase shift increases with the order of the filter.

Example: We want to make a unity gain low-pass filter that does not include oscillations either in the passband or in the attenuated band such that: $A_p = 1\text{db}$, $A_a = 40\text{db}$, $f_p = 1\text{kHz}$ and $f_a = 3\text{kHz}$.

Determine the filter and N.

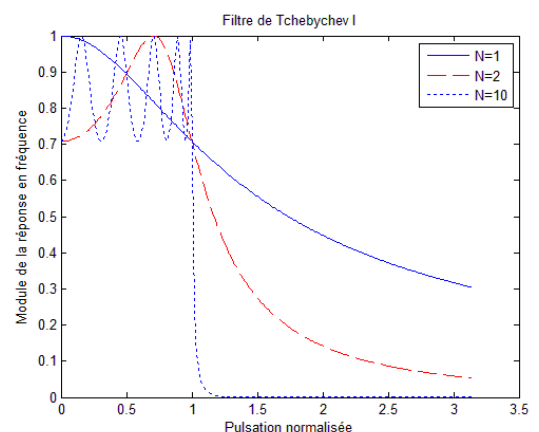
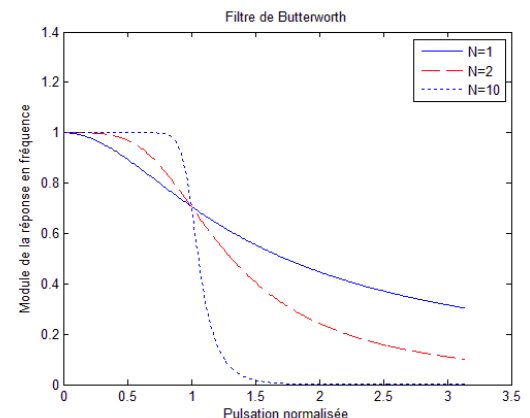
$$N \geq \frac{\log(10^4 / 0.2589)}{2 \log(3/1)} = 4.80 \approx 5$$

2. A Chebyshev filter is generally characterized by a balanced ripple in the cut band. The analytical form of the frequency response modulus of an analog Tchebychev (or Chebyshev) filter of order N is given by:

$$|H(\omega)| = \frac{1}{\sqrt{1 + \varepsilon^2 T_N^2\left(\frac{\omega}{\omega_c}\right)}}$$

Where ε is a real parameter less than unity which determines the amplitude of the oscillations in the passband and where $T_N(x)$ is the Chebyshev polynomial of order N, defined by:

$$T_N(x) = \begin{cases} \cos[N \arccos(x)] & \text{pour } |x| \leq 1 \\ \cosh[N \operatorname{arccosh}(x)] & \text{pour } |x| \geq 1 \end{cases}$$



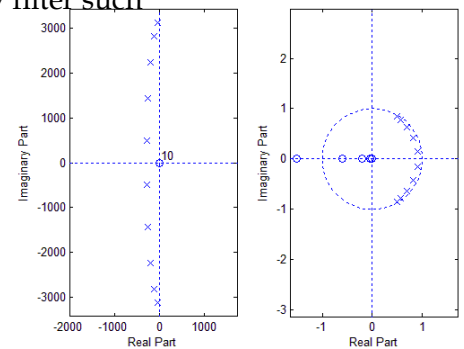
The order N is determined by the transition zone ($\omega_a - \omega_p$) and the ripples allowed in passband $A_p = 20 \log(1 + \delta_1)$ and attenuated band ($A_a = -\log \delta_2$) *

$$N \geq \frac{\log \left(\sqrt{\left(10^{\frac{A_a}{20}} - 1\right) / \left(10^{\frac{A_p}{20}} - 1\right)} + \sqrt{\left(10^{\frac{A_a}{20}} - 1\right) / \left(10^{\frac{A_p}{20}} - 1\right)} \right)}{\log \left(\omega_a / \omega_p + \sqrt{(\omega_a / \omega_p)^2 - 1} \right)} \quad N \geq \frac{\log \left(2.10^{\frac{A_a}{20}} / \sqrt{\left(10^{\frac{A_p}{20}} - 1\right)} \right)}{\log \left(\omega_a / \omega_p + \sqrt{(\omega_a / \omega_p)^2 - 1} \right)}$$

Example

We wish to produce a low-pass filter of unity gain with a Tchebyshev filter such that: $A_p = 1\text{dB}$, $A_a = 40\text{dB}$, $f_p = 1\text{kHz}$ and $f_a = 3\text{kHz}$. Determine N

$$N \geq \frac{\log \left(2.10^{\frac{A_a}{20}} / \sqrt{\left(10^{\frac{A_p}{20}} - 1\right)} \right)}{\log \left(\omega_a / \omega_p + \sqrt{(\omega_a / \omega_p)^2 - 1} \right)} = \frac{2.100 / 0.5089}{\sqrt{3 + \sqrt{8}}} = 3.39 \approx 4$$



In practice, three ripple values are used, $\epsilon = 0.1\text{ dB}$, 0.5 dB and 1 dB . For these 3 values $H_N(p)$ is given:

Ordre du filtre	Dénominateur (le numérateur est à 1)
2	$0.3017 p^2 + 0.7158 p + 1$
3	$(0.5918 p^2 + 0.5736 p + 1)(1.031 p + 1)$
Ordre du filtre	Dénominateur (le numérateur est à 1)
2	$0.9070 p^2 + 0.9956 p + 1$
3	$(1.0058 p^2 + 0.497 p + 1)(2.023 p + 1)$
4	$(1.0136 p^2 + 0.2828 p + 1)(3.5791 p^2 + 2.4113 p + 1)$
5	$(2.3293 p^2 + 1.0911 p + 1)(1.0118 p^2 + 0.1610 p + 1)(3.454 p + 1)$

Ordre du filtre	Dénominateur (le numérateur est à 1)
2	$0.6595 p^2 + 0.9402 p + 1$
3	$(0.8753 p^2 + 0.5483 p + 1)(1.596 p + 1)$
4	$(0.9402 p^2 + 0.3297 p + 1)(2.8057 p^2 + 2.3755 p + 1)$
5	$(2.0974 p^2 + 1.2296 p + 1)(0.9654 p^2 + 0.2161 p + 1)(2.759 p + 1)$

We recall that the transfer functions $H_N(p)$ of polynomial analog filters (Butterworth, Tchebyshev, Bessel, Cauer, etc.) are given for standardized cutoff frequencies and only for low-pass filters [2].

low pass	$p = p / \omega_A$
high pass	$p = \omega_A / p$

Bandpass	$p = \frac{1}{B} \left(\frac{p}{\omega_A} + \frac{\omega_A}{p} \right)$	Central pulse $\omega_A = \sqrt{\omega_{A1} \omega_{A2}}$
Tape cutter	$p = \left[\frac{1}{B} \left(\frac{p}{\omega_A} + \frac{\omega_A}{p} \right) \right]^{-1}$	Bandwidth $B = (\omega_{A2} - \omega_{A1})$

From the normalized expression of $H_N(p)$, we denormalize by replacing p with the values given in the previous table, resulting in a denormalized transfer function $H(p)$.

The objective of the bilinear transformation method is to make the analog and digital domains coincide as well as possible. The filters derived from it are more stable than are obtained through the use of the impulse variance method. But, on the other hand, it introduces a distortion on the frequency axis.

It is known that an analog filter is characterized by its transfer function $H(p)$ and a digital filter is defined by its transfer function $H(z)$ with $z = e^{2\pi j f T_e} = e^{p T_e}$. Which implies that $p = \ln(z)/T_e$. In both cases, this transformation results in a non-linear relationship between the frequencies f_A of the analog domain and the frequencies f_N of the digital domain.

To preserve the polynomial nature of the transfer functions, an approximation of $\ln(z)$ by the Laurent series is used:

$$\ln(z) \approx 2 \left(\frac{1-z^{-1}}{1+z^{-1}} + \frac{1}{3} \left(\frac{1-z^{-1}}{1+z^{-1}} \right)^3 + \frac{1}{5} \left(\frac{1-z^{-1}}{1+z^{-1}} \right)^5 + \dots \right)$$

Keeping only the first-order term, we define the bilinear transformation:

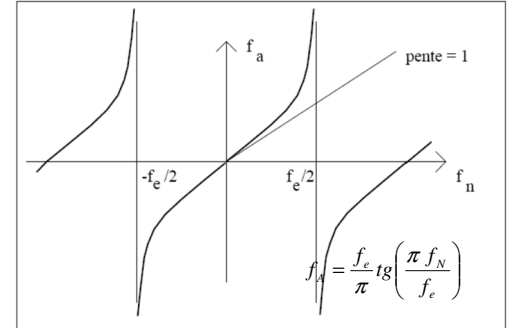
$$p = \frac{\ln(z)}{T_e} \approx \frac{2}{T_e} \frac{1-z^{-1}}{1+z^{-1}} = \frac{2}{T_e} \frac{z-1}{z+1}$$

$$\text{Knowing that } p = j\omega_A = \frac{\ln(z)}{T_e} = \frac{2}{T_e} \frac{e^{j\omega_N T_e} - 1}{e^{j\omega_N T_e} + 1} = \frac{2}{T_e} \frac{e^{j\omega_N T_e/2} (e^{j\omega_N T_e/2} - e^{-j\omega_N T_e/2})}{e^{j\omega_N T_e/2} (e^{j\omega_N T_e/2} + e^{-j\omega_N T_e/2})} = \frac{2}{T_e} \frac{j \sin(\omega_N T_e/2)}{\cos(\omega_N T_e/2)}$$

Thus, it can be shown that the correspondence between analog frequency f_A (or analog pulse ω_A) and digital frequency f_N (or ω_N) is obtained by:

$$\omega_A = \frac{2}{T_e} \operatorname{tg} \left(\frac{\omega_N T_e}{2} \right) = \frac{2}{T_e} \operatorname{tg} \left(\frac{\pi f_N}{f_e} \right)$$

This equation shows that there is no equality between analog pulsation and discrete pulsation and that the relation linking them is not linear either since there is distortion of the frequencies, including the group delay.



To synthesize a digital filter by bilinear transformation, proceed as follows:

- We define the desired characteristics of the digital filter (sampling frequency, cut-off, etc.)
- We calculate the analog pulsations ω_A corresponding to the digital pulsations $\omega_A = \frac{2}{T_e} \operatorname{tg} \left(\frac{\pi f_N}{f_e} \right)$
- We determine the template of the analog filter $H_N(p)$ normalized of order n (Chebyshev, Butterworth, etc.) which will serve as a model for the digital filter and we write the denormalized transfer function $H(p)$ of this analog filter (which must be recalculated according to ω_A).
- We apply the bilinear transformation to $H(p)$ by replacing, $p = \frac{2}{T_e} \frac{1-z^{-1}}{1+z^{-1}}$ which gives the function $H(z)$.

Example 1: We want to design a first-order digital low-pass filter from a transfer function of an RC filter in the continuous domain ($H_N(p) = 1/(1+p)$). The desired cutoff frequency is $f_N = 30$ Hz and the sampling frequency is $f_e = 150$ Hz.

We first calculate $\omega_A = \frac{2}{T_e} \operatorname{tg} \left(\frac{\pi}{5} \right) = \frac{2}{T_e} 0.7265$ and then $H(p) = \frac{1}{(1+p)} \Big|_{p=p/\omega_A} = \frac{\frac{2}{T_e} 0.7265}{p + \frac{2}{T_e} 0.7265}$

$$H(z) = H(p) \Big|_{p=\frac{2}{T_e} \frac{z-1}{z+1}} = \frac{\frac{2}{T_e} 0.7265}{\frac{2}{T_e} \frac{z-1}{z+1} + \frac{2}{T_e} 0.7265} = \frac{0.7265(z+1)}{(z-1) + 0.7265(z+1)} = \frac{0.7265(z+1)}{1.7265z - 0.2735}$$

It can be noticed that $f_A = \frac{2}{T_e(2\pi)} \operatorname{tg} \left(\frac{\pi}{5} \right) = \frac{f_e}{\pi} 0.7265 = 34.69 \neq f_N = 30$

Example 2 : We want to create a second-order low-pass digital filter with the following characteristics [18]: cut-off frequency $f_N = 500$ Hz (gain of 1), sampling frequency $f_e = 5$ kHz,

$$\omega_A = \frac{2}{T_e} \operatorname{tg} \left(\frac{\omega_N T_e}{2} \right) = \frac{2}{T_e} \operatorname{tg} \left(\frac{\pi}{10} \right) = \frac{2}{T_e} 0.325$$

$$H_N(p) = \frac{0.1075}{p^2 + 0.1075p + 1} \Rightarrow H(p) = H_N(p = p/\omega_A) = \frac{0.1075\omega_A^2}{p^2 + 0.1075p\omega_A + \omega_A^2} = \frac{\frac{4}{T_e^2} 0.0114}{p^2 + \frac{2}{T_e} 0.035p + \frac{4}{T_e^2} 0.1056}$$

$$H(z) = H(p) \Big|_{p=\frac{2}{T_e} \frac{z-1}{z+1}} = \frac{\frac{4}{T_e^2} 0.0114}{\left(\frac{2}{T_e} \frac{z-1}{z+1} \right)^2 + \frac{2}{T_e} 0.035 \frac{z-1}{z+1} + \frac{4}{T_e^2} 0.1056} = \frac{0.0114}{\left(\frac{z-1}{z+1} \right)^2 + 0.035 \frac{z-1}{z+1} + 0.1056}$$

$$H(z) = \frac{0.0114(z+1)^2}{(z-1)^2 + 0.035(z-1)(z+1) + 0.1056(z+1)^2} = \frac{0.0114(z+1)^2}{1.14z^2 - 1.7888z + 1.0706}$$

For a digital low-pass or high-pass filter, one can determine $H(z)$ directly from $H_N(p)$ using the normalized pulsation $\Omega_c = \operatorname{tg} \left(\frac{\pi f_N}{f_e} \right)$ and replace p in $H_N(p)$ as follows:

- for a low pass by $p = \frac{1}{\Omega_c} \frac{z-1}{z+1}$ either $H(z) = H_N(p) \Big|_{p=\frac{1}{\Omega_c} \frac{z-1}{z+1}}$

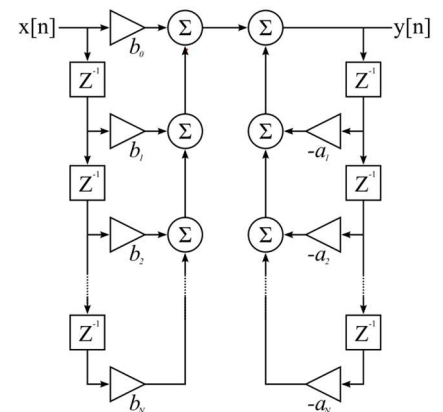
- for a high pass by $p = \Omega_c \frac{z+1}{z-1}$ either $H(z) = H_N(p) \Big|_{p=\Omega_c \frac{z+1}{z-1}}$

- for a pass band or a band rejector, the determination of ω_A is obtained by the central pulsation $\omega_A = \sqrt{\omega_{A1}\omega_{A2}}$

Note : Since it is not possible to precisely match the frequency response of the analog filter (on $[0, +\infty[$ and digital (on $[0, f_e/2]$) the bilinear transformation will aim to match as well as possible both answers. This method gives fairly good results provided that the characteristic frequencies are not too close to the half-sampling frequency.

• Realization of the IIR filter

An IIR filter requires $(2N+1)$ multiplication operations, $2N$ addition operations for each new sample to be filtered or $(2N+1)$ MAC (Multiplication-Accumulation). The memory cost of an IIR filter in direct structure is $(4N+3)$ memory [($2N+1$) coefficients b_i and $2(N+1)$ memory points or delay for the vectors of the inputs $x(n)$ and the outputs $y(n)$] [9].



7. Multi-rate filters

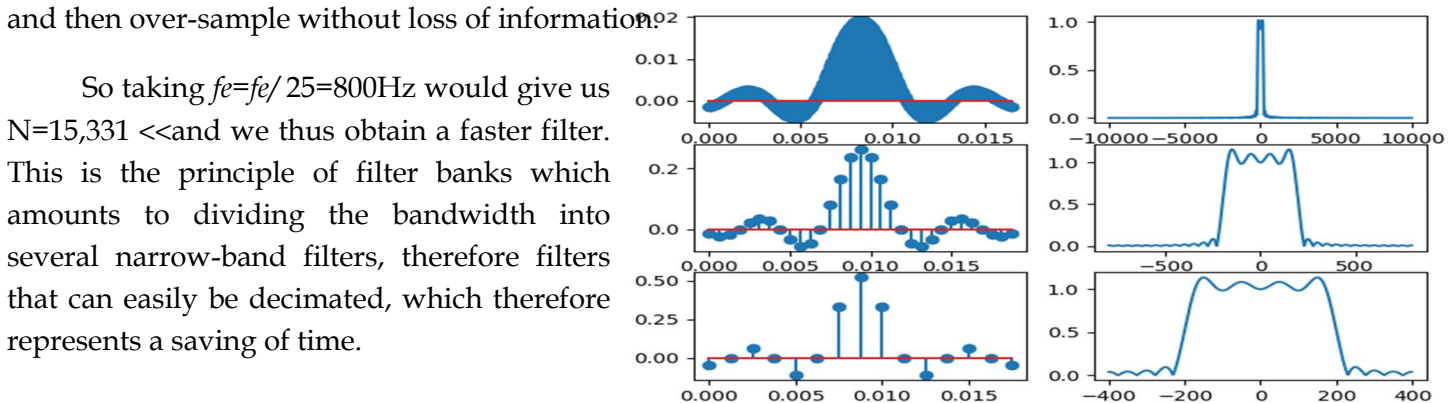
Multi-rate processing involves multiple sample rates. It can have several motivations. It may be the result of constraints related to an application in which the digital signal streams to be processed and generated must be at different sampling frequencies [20]. This situation can be encountered for example in audio applications where several sampling frequencies co-exist: 32kHz (diffusion); 44.1 kHz (digital compact disc); 48kHz (digital soundtrack); 96kHz; ...

This approach can also make it possible to improve the implementation characteristics of an algorithm by adopting for each of its steps a sampling which is called critical in the sense that the processing frequency is chosen equal to the Nyquist frequency. Thus the processing chain corresponding to the baseband transmitter of a DPQSK modulator where there are three processing frequencies: the bit frequency, the symbol frequency and the sampling frequency [20]

In the systems considered in the previous chapters, only a single frequency or sampling rate $f_e = 1/T_e$ was considered. But it happens that the maximum frequency f_{max} is decreased (low-pass filtering) or increased (the case of a modulation). In both cases, it is more judicious to adapt the sampling frequency in order to minimize the calculation time. It will be a decimation or under-sampling (increase of f_{max}) and an interpolation or over-sampling (reduction of f_{max}). Also note that oversampling is often used to increase SNR in DAC operations. Both are involved in frequency shifting and filter bank techniques

Example: Suppose we want to synthesize a low-pass filter with the following specifications: $f_p = 100\text{Hz}$, $f_a = 300\text{Hz}$, $A_a > 50\text{ dB}$, $f_e = 20\text{kHz}$.

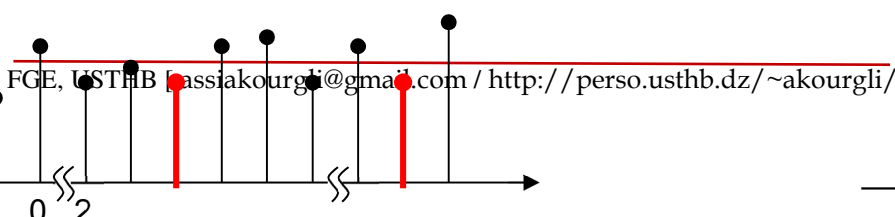
The use of the Hamming window gives us $N=331$, ie a large number of coefficients. However, the filter occupies a small portion of the bandwidth (low-pass with $f_c = 1\% f_s$). It is therefore possible to under-sample



Undersampling and oversampling

Recall that when we wish to sample a signal, we must respect Shannon's theorem, namely take a sampling frequency $f_e / 2 > f_{max}$. If this signal has undergone processing which has resulted, among other things, in the modification of the maximum frequency. We will then speak of under-sampling or over-sampling.

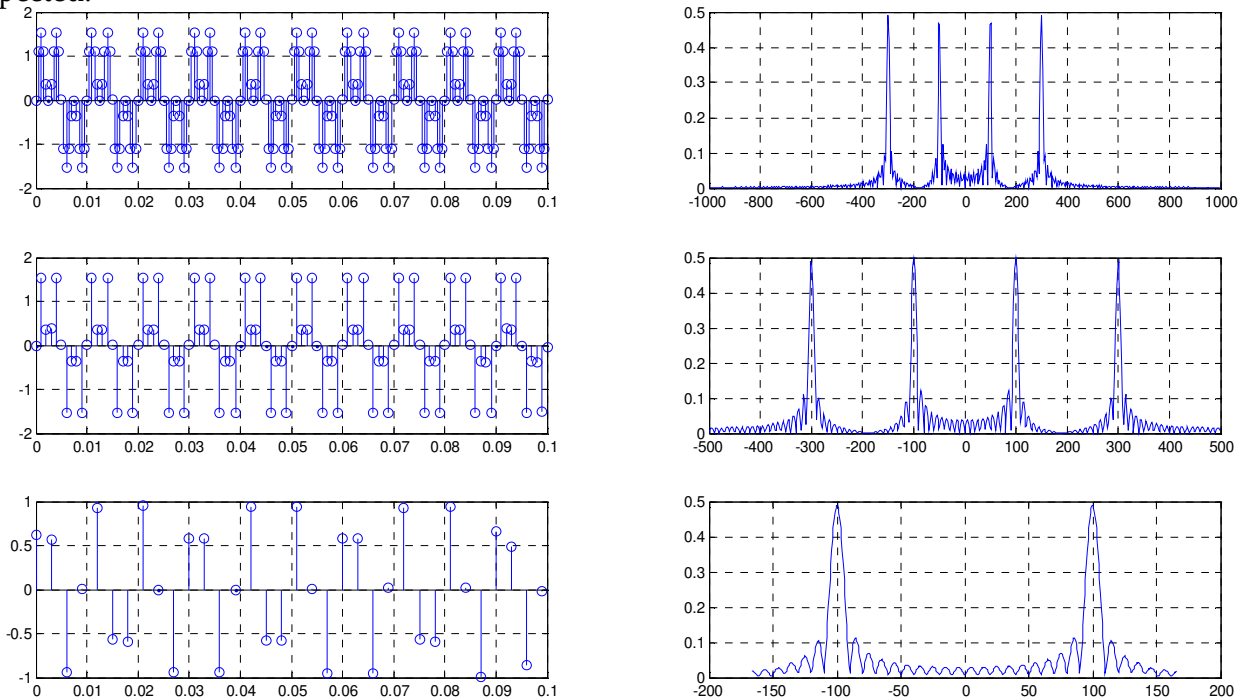
Subsampling : It consists of manufacturing from the original signal x a signal x_D comprising fewer samples than the original signal. It is necessary to "remove" samples, hence the name decimation (In English: down sampling). We speak of undersampling by a factor D when the original signal x has N samples and the oversampled signal x_D has N/D , with $x_D(n) = x(n/D)$



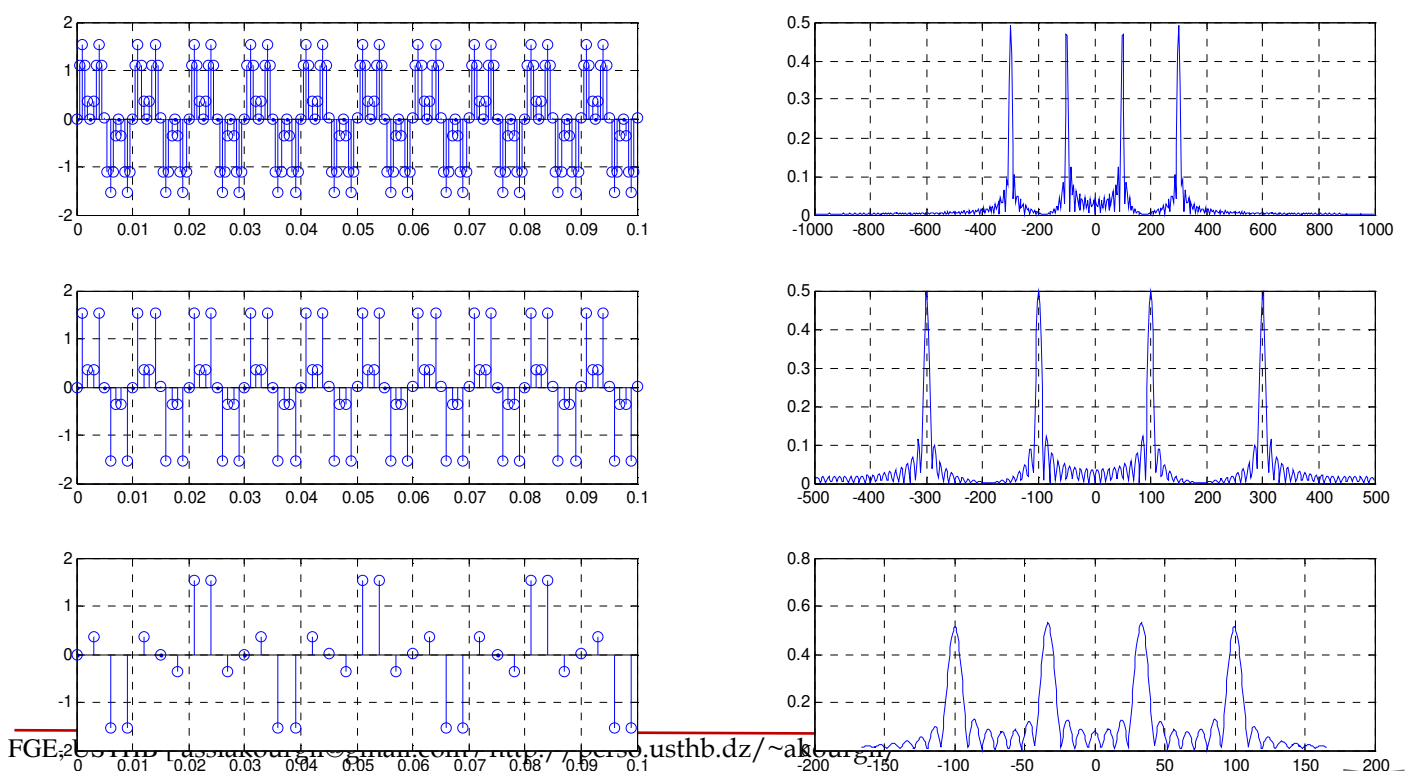
$$x(n) \rightarrow \downarrow_D \rightarrow x_D(m)$$

Naive subsampling involves getting rid of $D - 1$ samples out of D . This amounts to having to sample directly at f_e/D , but there is no guarantee that $f_e/D > 2f_{max}$. There is therefore a risk of spectral aliasing.

Example : The following signal is the result of the sum of 2 sinusoids with frequencies 100 and 300Hz. We decimate it by 2 and 6. We notice that when $f_e = f_e/6$ there is a folding since Shannon's condition is no longer respected.



To downsample correctly, it will be necessary to remove the frequency content of $x(n)$ between $f_e/2D$ and f_e/D since the spectrum becomes periodic with period f_e/D and this before decimation.



In this case, we have for the transfer function $X_D(z) = \sum_{-\infty}^{+\infty} x_D(m)z^{-m} = \sum_{-\infty}^{+\infty} x(n/D)z^{-n/D}$

Reminders:

Dirac comb: $\sum_{n=-\infty}^{\infty} \delta(t - nT_e)$ periodical with period $T_e \Rightarrow$ decomposable in Fourier series such that

$$C_n = \frac{1}{T_e} \Rightarrow \sum_{n=-\infty}^{\infty} \delta(t - nT_e) = \frac{1}{T_e} \sum_{n=-\infty}^{\infty} e^{2\pi j n \frac{t}{T_e}} \quad \text{In discrete } \sum \delta(n - mM) = \frac{1}{M} \sum_{m=0}^{M-1} e^{2\pi j m \frac{n}{M}}$$

Decimating M is equivalent to sampling $M \Rightarrow y(m) = y\left(\frac{n}{M}\right) = \sum x(n) \cdot \delta(n - mM)$

$$\blacksquare \text{ Transfer function } Y(z) = \sum_{n=-\infty}^{\infty} \left(x(n) \frac{1}{M} \sum_{k=0}^{M-1} e^{\frac{2\pi j k n}{M}} \right) z^{-n/M} = \frac{1}{M} \sum_{k=0}^{M-1} \left(\sum_{n=-\infty}^{\infty} x(n) \left(e^{-\frac{2\pi j k}{M}} z^{\frac{1}{M}} \right)^{-n} \right)$$

$$\Rightarrow Y(z) = \frac{1}{M} \sum_{k=0}^{M-1} X\left(W_M^k z^{\frac{1}{M}}\right) \text{ with } W_M = e^{-\frac{2\pi j}{M}}$$

Frequency response

$$y(m) = x(n) \cdot \sum \delta(n - mM) \Rightarrow Y(f) = X(f) * \frac{f_e}{M} \sum_{m=0}^{M-1} \delta\left(f - \frac{mf_e}{M}\right)$$

$$\Rightarrow Y(f) = \frac{f_e}{M} \sum_{m=0}^{M-1} X\left(f - \frac{mf_e}{M}\right) \Rightarrow \text{Periodization de } \frac{f_e}{M}$$

Examples:

- Decimation of 4 $Y(z) = \frac{1}{4} \sum_{k=0}^3 X\left(W_4^k z^{\frac{1}{4}}\right) \text{ with } W_4 = e^{-\frac{2\pi j}{4}}$

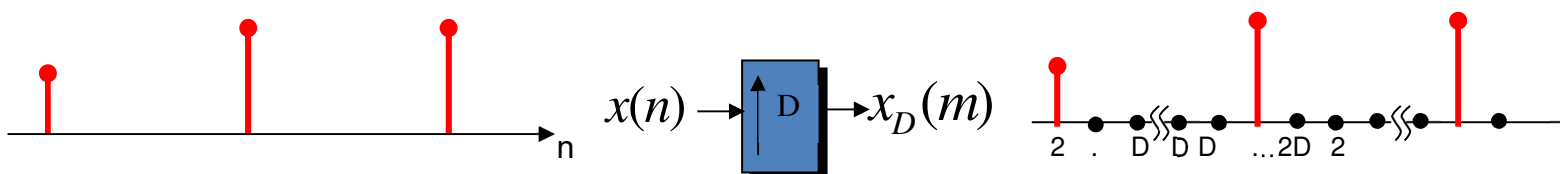
$$Y(z) = \frac{1}{4} \left(X\left(W_4^0 z^{\frac{1}{4}}\right) + X\left(W_4^1 z^{\frac{1}{4}}\right) + X\left(W_4^2 z^{\frac{1}{4}}\right) + X\left(W_4^3 z^{\frac{1}{4}}\right) \right)$$

$$Y(z) = \frac{1}{4} \left(X\left(z^{\frac{1}{4}}\right) + X\left(-j z^{\frac{1}{4}}\right) + X\left(-z^{\frac{1}{4}}\right) + X\left(j z^{\frac{1}{4}}\right) \right) \Rightarrow Y(f) = \frac{1}{4} \sum_{k=0}^3 X\left(f - \frac{kf_e}{4}\right)$$

o Decimation of 2

$$Y(z) = \frac{1}{2} \sum_{k=0}^1 X\left(W_2^k z^{\frac{1}{2}}\right) \text{ with } W_2 = e^{-\frac{2\pi j}{2}} \quad Y(z) = \frac{1}{2} \left(X\left(z^{\frac{1}{2}}\right) + X\left(-z^{\frac{1}{2}}\right) \right) \Rightarrow Y(f) = \frac{1}{2} \left[X(f) + X\left(f - \frac{f_e}{2}\right) \right]$$

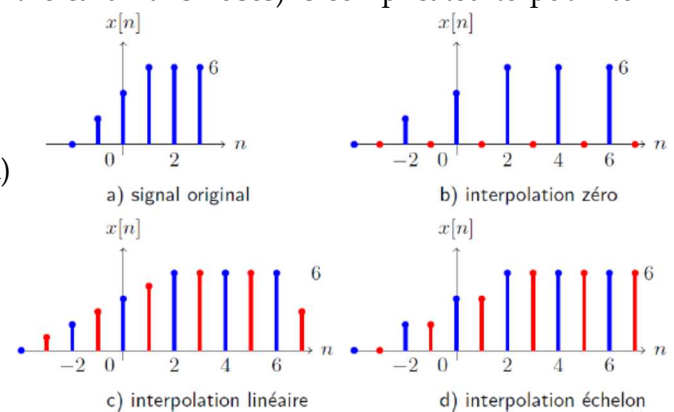
Over-sampling: It consists in manufacturing from the original signal x a signal x_M a signal of the same duration but comprising more samples than the original signal. It is necessary to "add" samples, hence the name of interpolation (in English: upsampling). One speaks of oversampling by a factor M when the original signal x comprises N samples and the oversampled signal comprises MN , with $x_M(Mn) = x(n)$.



Theoretically, if the original signal was well sampled, perfect reconstruction is possible, so too is oversampling. However, the reconstruction formula (with the cardinal sinuses) is complicated to put into practice. We appeal to simpler ideas [25]:

- Interleave zeros (zero interpolation)
- Copy the value of the previous time (step interpolation)
- Draw a line between two available samples

Adjacent (linear interpolation)



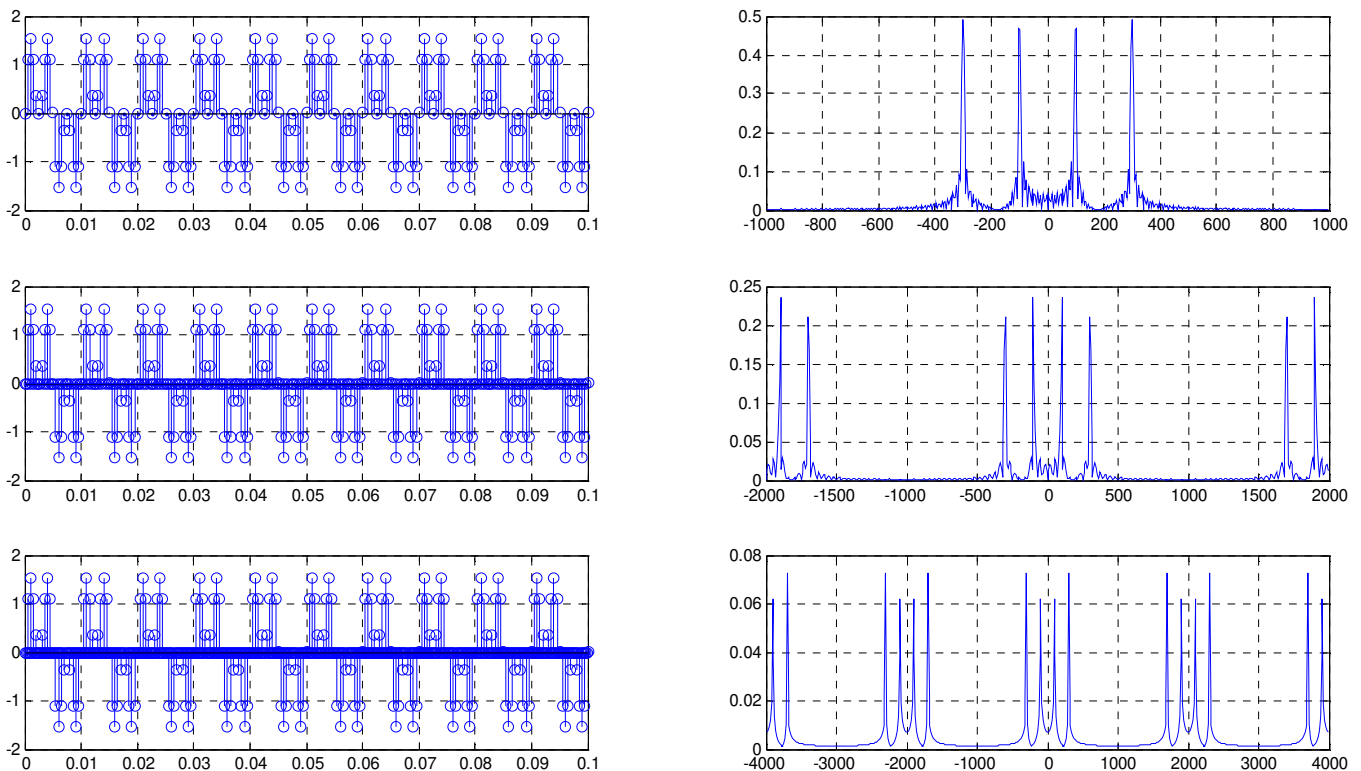
The original signal has no frequency content beyond $f_e/2$. The interpolation process is likely to add more. It is therefore followed by low-pass filtering at the cutoff $f_e/2$ to remove the added content. This is the reason why the interpolation method does not matter much.

Application exercise: VeFIRy that the zero interpolation preserves the signal spectrum.

$$X_D(z) = \sum_{-\infty}^{+\infty} x_D(m)z^{-m} = \sum_{-\infty}^{+\infty} x_D(nD)z^{-nD} = \sum_{-\infty}^{+\infty} x(n)(z^{-n})^D = X(z^D)$$

Which in frequency gives us: $X_D(f) = X(z = e^{2\pi j f D T_e} = e^{2\pi j f D / f_e})$ which means that the original and oversampled signal have the same spectrum, we only go through the circle D times.

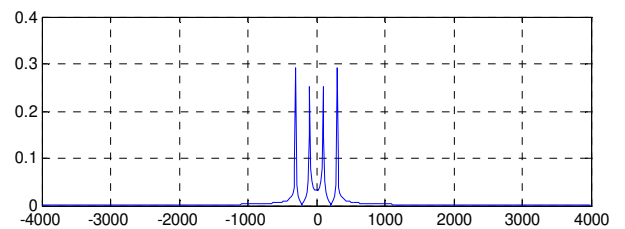
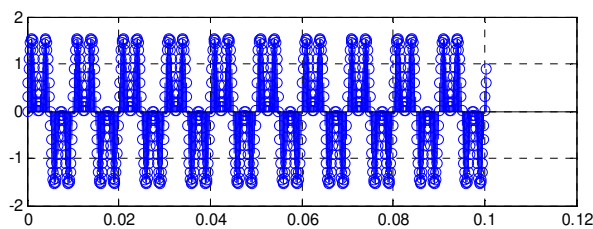
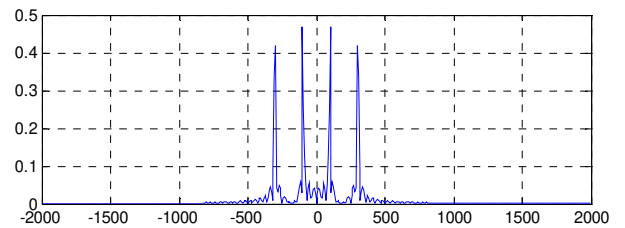
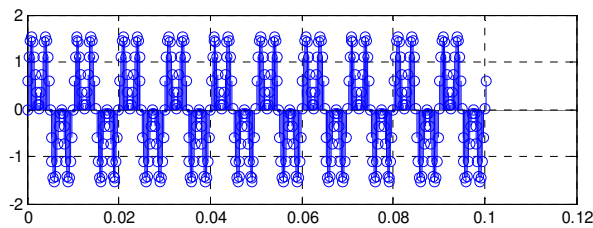
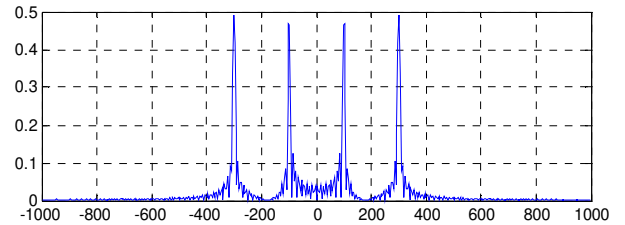
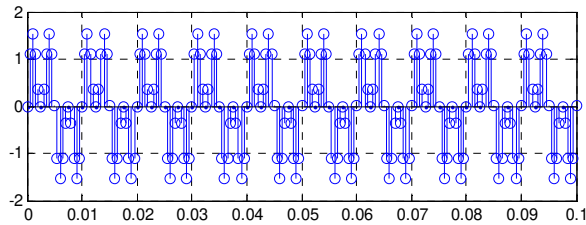
Example: We take the same signal and interpolate by 2 then by 4.



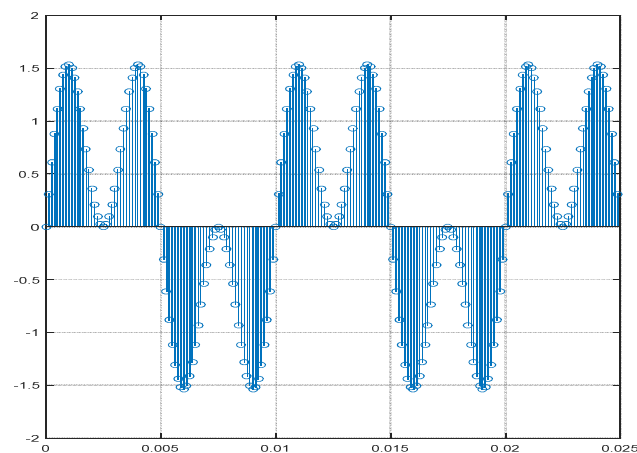
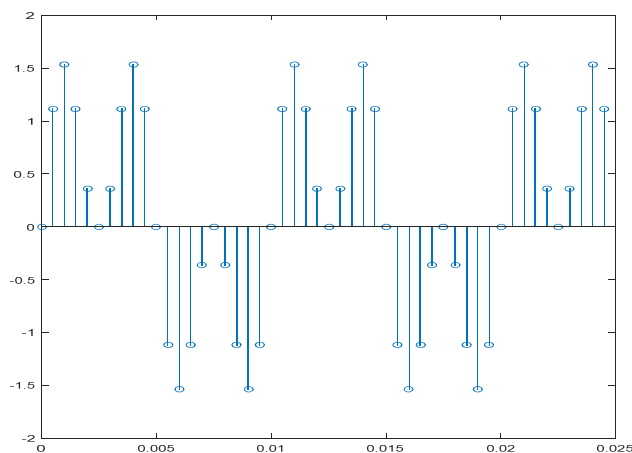
After upsampling, the shape of the spectrum remains the same (no energy has been added to the

signal), but the sampling frequency is now Df_e . Oversampling causes what are called mirror spectra to appear and to obtain a signal whose temporal form would correspond to the signal $x(n)$ which would have been sampled at a frequency Df_e .

It is necessary to remove these mirror spectra and therefore filter by a low-pass filter the over-sampled signal $y(m)$ with a filter having an attenuated band from $f_e/2$



We can observe the effect of interpolation followed by anti-mirror low-pass filtering (equivalent to 0 padding in the frequency domain)



Examples:

- Interpolation de 4 $Y(z) = X(z^4)$
- Interpolation de 2 $Y(z) = X(z^2)$

Noticed : Polyphase decomposition occurs in filter banks, it allows a simple representation for over and under sampling [21]. When a signal undergoes a decimation by a factor of 2, this amounts to separating the samples with an even index from those with an odd index, or again, to separating two distinct phases having different delays (also called phase deviations) in the vector [22]. If we consider a decimation of a factor M, the principle is exactly the same but with M distinct phases.

Rational frequency shift

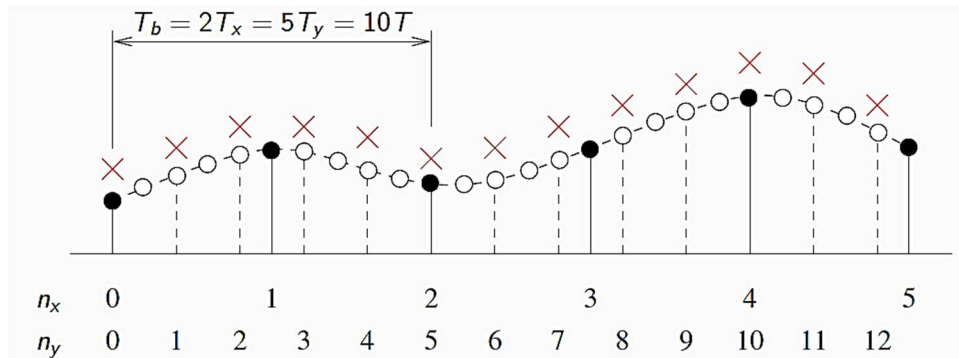
Suppose that we want to digitally change the sampling frequency of a signal $x(n)$ and go from a sampling frequency f_1 to a sampling frequency f_2 where $f_2/f_1 = L/M$. This therefore leads to over-sampling the signal $x(n)$ by a factor L and then to decimating it by a factor M.

- The over-sampling by L must be followed by a low-pass filter cutting at $f_c/2L$ to remove the spectra due to over-sampling where f is the sampling frequency of the output signal.
- The decimation by M must be preceded by an anti-aliasing filtering having a cut-off frequency of $f_c/2M$, if f_c is the sampling frequency of the input signal.

Only one of the two low-pass filters will then be kept: the one whose cutoff frequency is the lowest



Note: If Let M are coprime, decimation and interpolation are commutative, below is an example: of conversion where $L = 5$ and $M = 2$ [31]



Moreover, according to number theory, if L and M are coprime then there exist two integers l_0 et m_0 that $l_0L - m_0M = -1 \rightarrow z^{-1} = z^{-(l_0L - m_0M)} = z^{-l_0L} z^{m_0M}$

Example: We want to reduce the sampling frequency of an audio signal by a factor of 2/3 (from 48kHz to 32kHz). Note that decimation and interpolation are not commutative unless L and M are relatively prime [20].

Notes _

A very narrow bandwidth decimation/interpolation filter is complex to implement (High Order)

. It can be replaced by the cascading of simpler filters. Individual filter specifications are relaxed considerably since the overall filter specification is shared among multiple lower order filters.

Thus, for a decimation or interpolation factor, it is more judicious to operate in stages (multi-stages).

If we have to decimate by 15, this requires a high order filter. We can then proceed by decimating by 3 then by 5. This will allow us to use 2 filters of lower order.

The applications of multi-rate processing are numerous: Design of phase shifters, interfacing of digital systems at different rates, implementation of narrow-band LP filters, implementation of digital filter banks, sub-band coding of speech signals, quadrature mirror filters , transmultiplexers, oversampling in CA/N and CN/A.

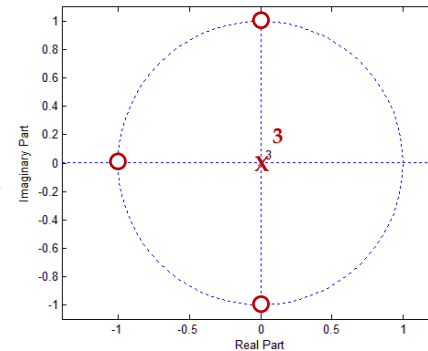
Exercices n°1: Digital Filters Synthesis and Multi-rate Filtering

1. Let $H(z)$ be the transfer function of a causal LTI with: $H(z) = \frac{az-1}{z-a}$ with a real.

Determine the values of a for which $H(z)$ corresponds to a stable system. Take a value of $a = 0.5$. Then represent the poles and zeros of the function, the region of convergence. Give and Trace $|H(f)|$. Is this filter minimum phase?

2. Suppose that the plot of the poles and zeros of this system is as follows:

- Is it an FIR or IIR filter? (justify your answer)
- Give the approximate shape of $H(f)$
- Determine $H(z)$ then determine and plot $h(n)$
- From $h(n)$, study the stability, the causality and the invariance of this filter.
- Calculate and plot $H(f)$ for at least 3 values
- Does this filter have a constant group delay (justify)
- Determine its response for a step input $x(n) = U(n)$.



3. We consider that the recurrence equation of the following system is given as follows:

$$y(n) = x(n) - x(n-2) - 0.878 y(n-2)$$

- Study the causality and the invariance of this system
- Is it an FIR or IIR filter?
- Determine $H(z)$ and plot the poles and zeros, deduce the role of this filter:
- Give the approximate shapes of $h(n)$ and $|H(f)|$
- Determine the impulse response $h(n)$ and plot it for the first 3 values
- We assume that $f_e = 6$ kHz, what will be the output of the filter if we give as input: a signal noisy with a sinusoid of 500 Hz then a signal composed of 2 sinusoids, one of 1500 Hz and the other of 2500 Hz

4. Design a low-pass FIR filter by the windowing method to obtain the following specifications:

- Ideal cutoff frequency: $f_c = 1.75$ kHz
- Transition width: $\Delta f = 0.5$ kHz
- Attenuation in attenuated band: $A = -20 \log_{10}(\delta) > 51$ dB with $\delta = \min(\delta_1, \delta_2)$
- Sampling frequency: $f_e = 8$ kHz

5. We wish to approach an ideal high-pass filter by a finite impulse response filter, synthesized by the windowing method. This filter must meet the following specifications:

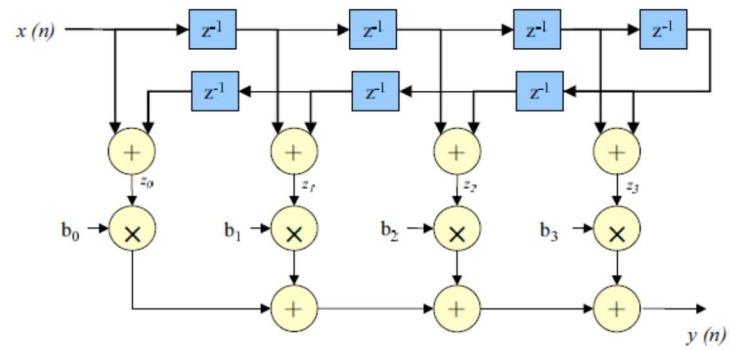
- ✓ Cutoff frequency $f_c = 2$ kHz
- ✓ Transition width: $\Delta f = 0.5$ kHz
- ✓ Attenuation in attenuated band: $A = -20 \log_{10}(\delta) > 40$ dB with $\delta = \min(\delta_1, \delta_2)$
- ✓ Sampling frequency: $f_e = 8$ kHz
- Determine the exact mathematical expression for $h'(n)$.
- Calculate $h'(0)$ and approximately plot $h'(n)$.
- What is the advantage of windowing and what is its disadvantage?
- What is the disadvantage of this filter synthesis technique?
- List an advantage and a disadvantage of synthesis by FIR filters as well as by IIRs.

6. We consider an FIR filter of length $N = 8$ whose

Structure is given opposite:

With $b_0=0.2$, $b_1=0.3$, $b_2=0.4$, $b_3=0.5$

- Determine the difference equation
- Determine the transfer function then $h(n)$
- Then identify the type of FIR filter (I, II, III or IV)
- What is its disadvantage?



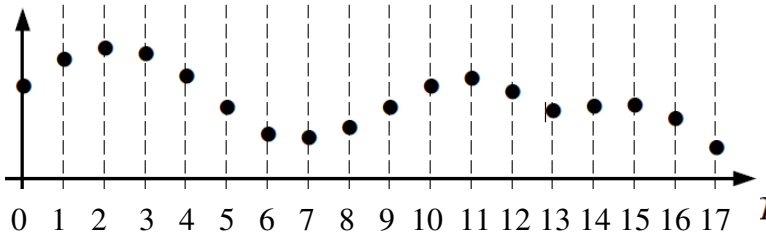
7. We consider the normalized single-pole analog low-pass filter $H_N(p)=1/(p+1)$. It is considered that the cutoff frequency normalized to -3db occurs for $f_c=f_e/10$. By bilinear transformation, find the equivalent low-pass digital filter $H(z)$.

8. Consider the following denormalized low-pass transfer function $H(p): (p+0.1)/((p+0.1)^2 + \omega_a^2)$. Knowing that $\omega_a=4$, use the bilinear transform method to transform this filter into a digital one. The digital filter will have its resonance for $\omega_N = \pi/(2.T_e)$.

9. Let the signal $x(n)=n U(n)$, give and plot the resulting signal if:

- We decimate it by 2, by 3 and by 4
 - We interpolate it by 2, by 4
 - We decimate it by 2 then we interpolate it by 2 and if we reverse the 2 operations
 - We decimate it by 3 and we interpolate it by 2 and if we reverse the 2 operations
- Check that decimation is not invariant to translation contrary to interpolation;

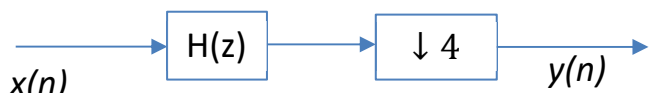
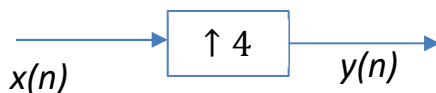
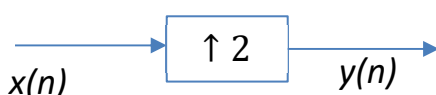
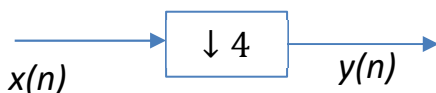
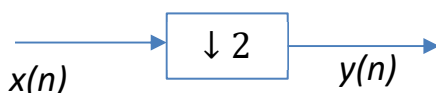
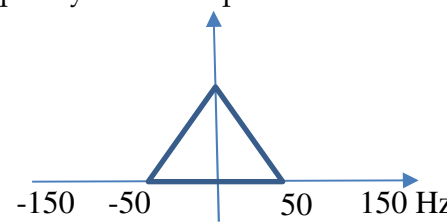
10. Consider the following signal, we want to decimate it by 2 then interpolate it by 2 as well



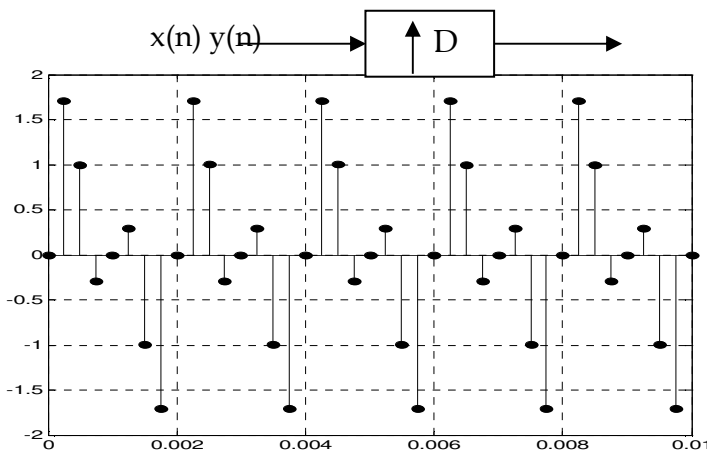
- Give the plot of the decimated then interpolated signal
- We consider that the frequency of the original signal is 10kHz, give the frequency at the output of the decimator then that of the interpolator.

11. We consider the signal $x(n)$ whose DTFT is given as opposite,

- Plot the DTFT of the corresponding output for the 8 cases:
- Give the cutoff frequency of the low-pass filter $H(z)$ for each case
- Specify the role of $H(z)$ in each case



12 . Before converting a digital signal $x(n)$ (below signal and its DTFT) , into an analog signal, we decide to interpolate it by D.



- What is the interest of interpolation in this case?
- Determine the expression of the signal $x(n)$.
- Plot the interpolated signal for $D=3$ and its DTFT.
- We consider that $D=6$, plot again the interpolated signal and its DTFT.
- Give the initial and final polyphase decomposition for $D=3$
- Plot the signal and its DFT in the case where the interpolation is followed by a low-pass filter at $f_c/2$

13 . An audio signal is recorded was transmitted at a frequency of 30 kHz, on reception before converting it to analog, a frequency change is made such that the new frequency is 45Hz.

- Give the general scheme of this operation using decimation and interpolation
- Give the cutoff frequency of each low-pass filter, which should be kept

Solutions

1 . The module is always worth 1 (all-pass cell), Maximum of phase

2. All poles at 0 then FIR, $H(z)=z^{-3}+z^{-2}+z^{-1}+1$, $h(n)=\delta(n-3)+\delta(n-2)+\delta(n-1)+1$,

$H(f)=2(\cos(3\pi fT_e)+\cos(\pi fT_e))e^{-3\pi j fT_e}$, Group delay cst, $x(n)=U(n)$ then $y(n)=U(n-3)+U(n-2)+U(n-1)+U(n)$

3. Causal, stable, IIR, notch filter, $H(z)=\frac{K(z^2-1)}{z^2+0.878}=\frac{K(1-z^{-2})}{1+0.878z^{-2}}$

4. $\Delta f=0.125 f_c=0.4375$ $h(n)=f_c \frac{\sin(\pi n f_c)}{\pi n f_c}=\frac{\sin(\pi n 0.4375)}{\pi n}$ for $k \neq 0$ and $h(0)=f_c$ Hamming $\Rightarrow N=53$

The filter coefficients are symmetrical, it will therefore suffice to calculate the values from $h(0)$ to $h(26)$.

$h(0)=f_c=0.4375$ $w(0)=0.54+0.46\cos(0)=1$ $h(0)=hN(0)$ $w(0)=0.4375$

$$w_{Ham}(n)=\begin{cases} 0.54+0.46\cos(\frac{2\pi n}{N-1}) & \text{pour } |n| \leq \frac{N-1}{2} \\ 0 & \text{ailleurs} \end{cases} = \begin{cases} 0.54+0.46\cos(\frac{2\pi n}{52}) & \text{pour } |n| \leq 26 \\ 0 & \text{ailleurs} \end{cases}$$

$h(0)=0.4375$, $h(1)=h(-1)=0.311$, $h(2)=h(-2)=0.060$, $h(3)=h(-3)=-0.0856$, $h(4)=h(-4)=-0.053$, $h(5)=h(-5)=0.0325$, $h(6)=h(-6)=0.0434$, $h(7)=h(-7)=-0.0075$, $h(8)=h(-8)=-0.0319$, , $h(26)=h(-26)=-0.0009$.

To make the filter causal, we add 26 to each of the indices.

5 . $\Delta f=0.5 f_c=0.125$ $h(n)=-f_c \frac{\sin(\pi n f_c)}{\pi n f_c}$ for $k \neq 0$ and $h(0)=1-f_c$ $A>40$ Hanning $\Rightarrow N=51$

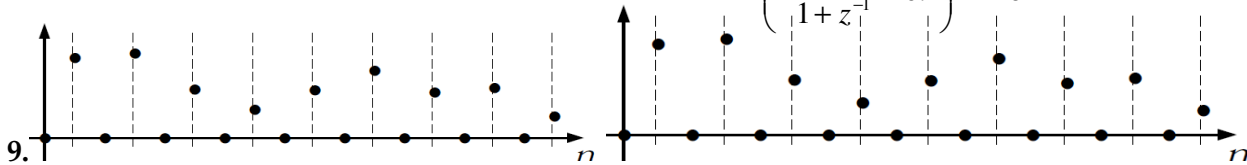
6. $y(n)=b_3(x(n-3)+x(n-4))+b_2(x(n-2)+x(n-5))+b_1(x(n-1)+x(n-6))+b_0(x(n)+x(n-7))$

Type II, not suitable for a high pass (a zero in -1)

$$7. \omega_A = \frac{2}{T_e} \operatorname{tg} \left(\frac{\pi}{10} \right) = \frac{0.65}{T_e} \Rightarrow H(p) = H_N(p) \Big|_{p/\omega_A} = \frac{\omega_A}{p + \omega_A} = \frac{0.65/T_e}{p + 0.65/T_e}$$

$$p = \frac{2}{T_e} \frac{1-z^{-1}}{1+z^{-1}} \Rightarrow H(z) = H(p) \Big|_{p=\frac{2}{T_e} \frac{1-z^{-1}}{1+z^{-1}}} = \frac{0.65/T_e}{\frac{2}{T_e} \frac{1-z^{-1}}{1+z^{-1}} + 0.65/T_e} = \frac{0.245(1+z^{-1})}{1-0.509z^{-1}}$$

$$8. \omega_a = 4 = \frac{2}{T_e} \operatorname{tg} \left(\frac{\pi}{4} \right) \Rightarrow T_e = 0.5 \Rightarrow p = 4 \frac{1-z^{-1}}{1+z^{-1}} \Rightarrow H(z) = \frac{4 \frac{1-z^{-1}}{1+z^{-1}} + 0.1}{\left(4 \frac{1-z^{-1}}{1+z^{-1}} + 0.1 \right)^2 + 16} = \frac{0.125 + 0.006z^{-1} - 0.118z^{-2}}{1 + 0.0006z^{-1} - 0.950z^{-2}}$$

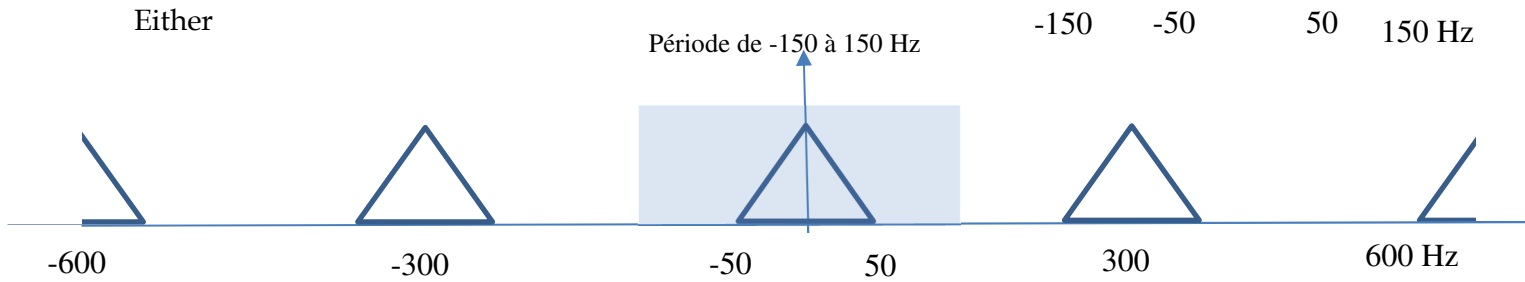


After decimation $f_e = 5 \text{ kHz}$

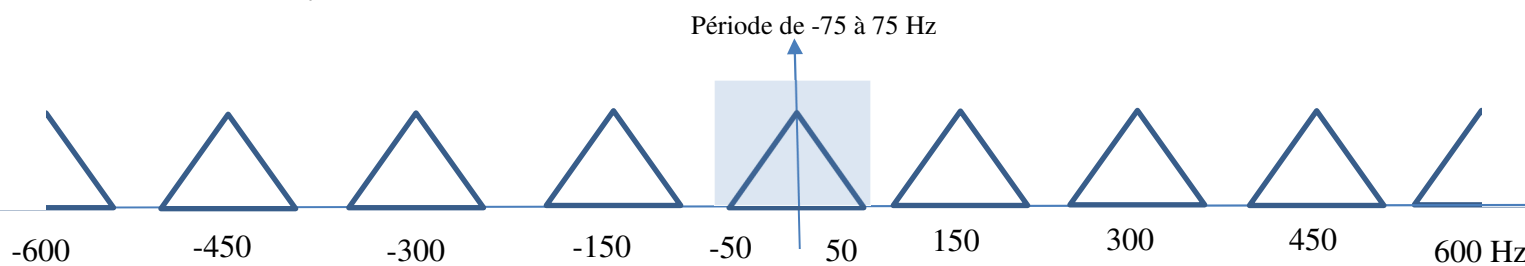
After interpolation $f_e = 10 \text{ kHz}$

10. The original signal is periodic with a period of 300 Hz

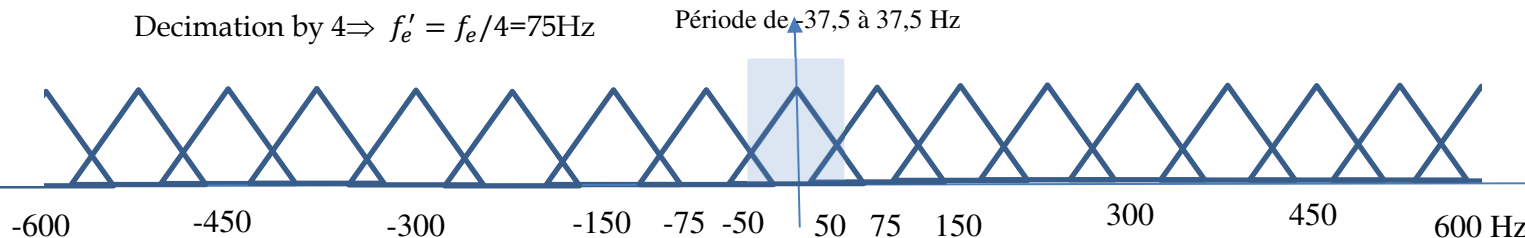
Either



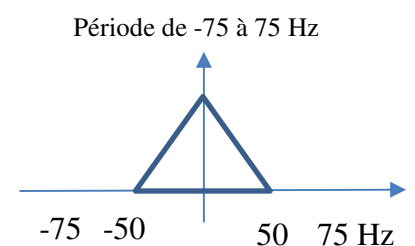
Decimation by 2 $\Rightarrow f'_e = f_e/2 = 150 \text{ Hz}$

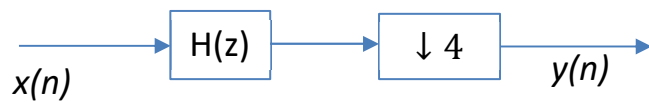


Decimation by 4 $\Rightarrow f'_e = f_e/4 = 75 \text{ Hz}$

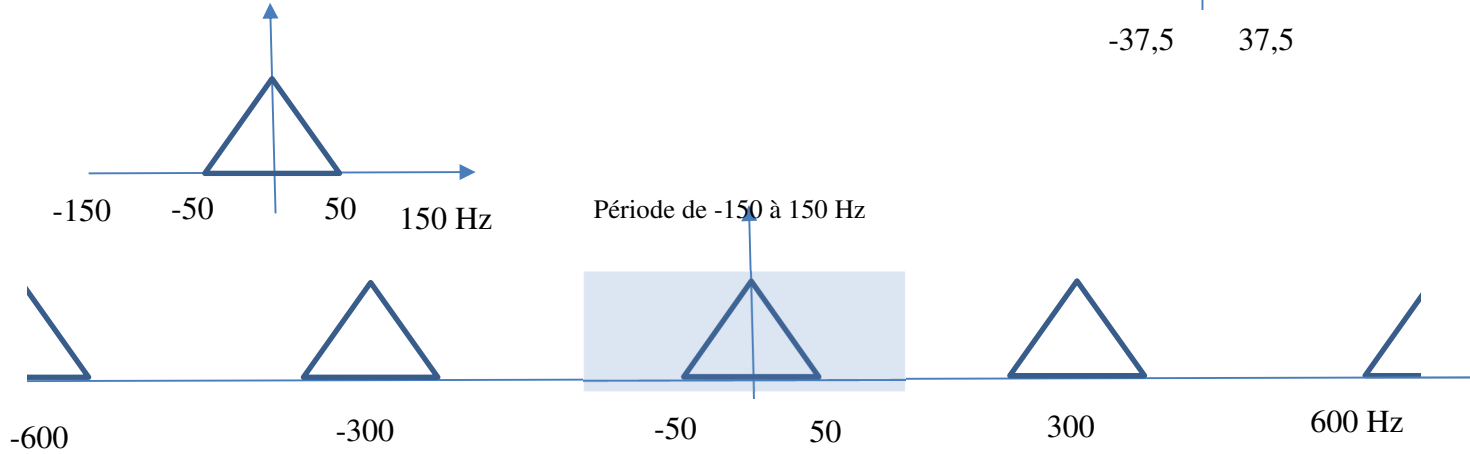
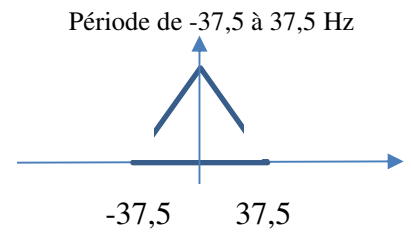


This is why we place a low pass filter at $f'_e/2 = f_e/(2M)$

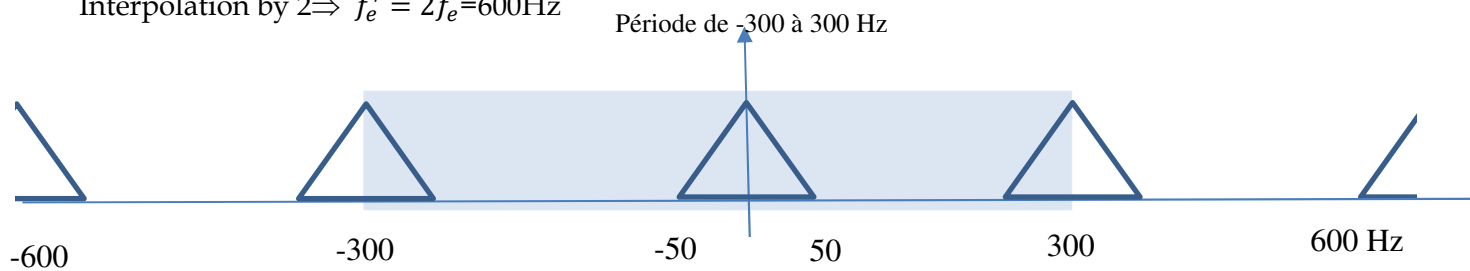




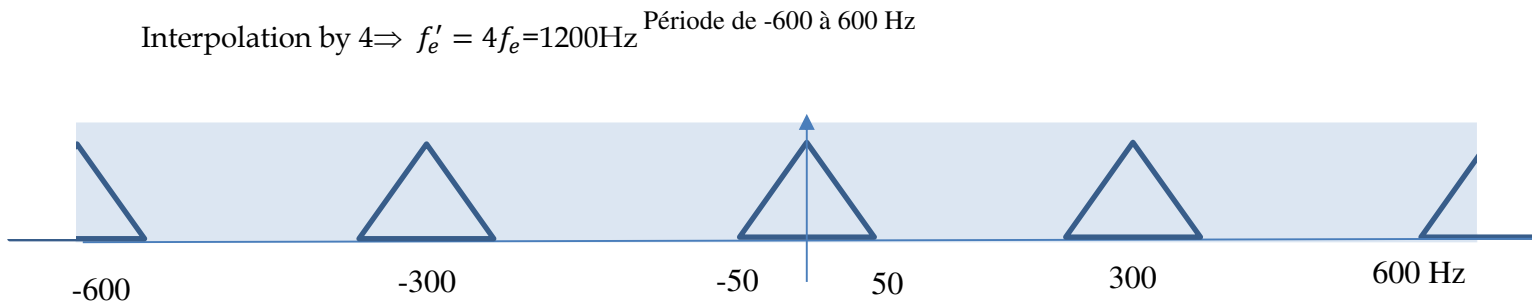
The original signal is periodic with a period of 300Hz



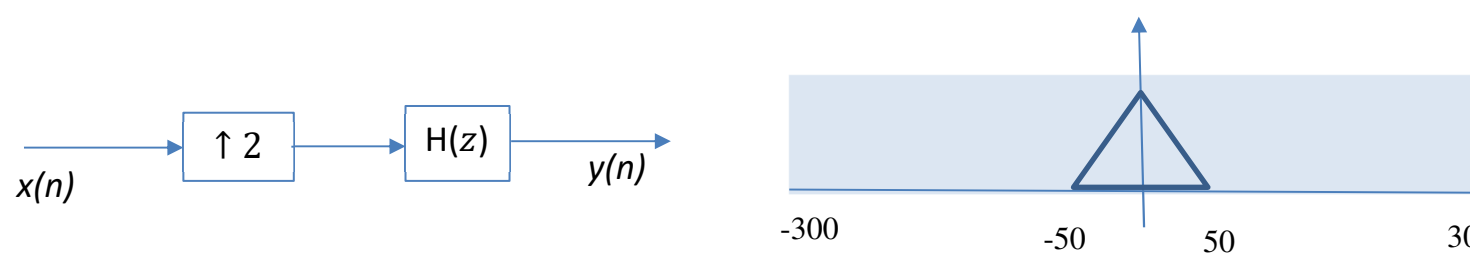
Interpolation by 2 $\Rightarrow f'_e = 2f_e = 600\text{Hz}$

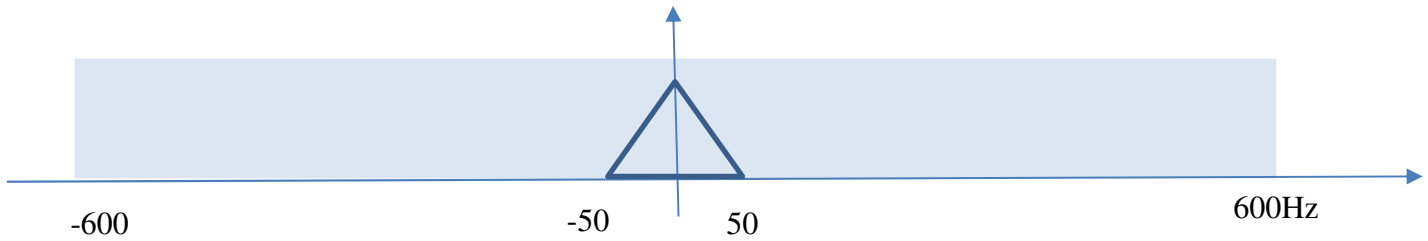


Interpolation by 4 $\Rightarrow f'_e = 4f_e = 1200\text{Hz}$



Appearance of mirror spectra \Rightarrow Low-pass filter at $f_e/2$





$$f_c = \frac{300}{2.2} = 75, f_c = \frac{300}{2.4} = 37.5$$

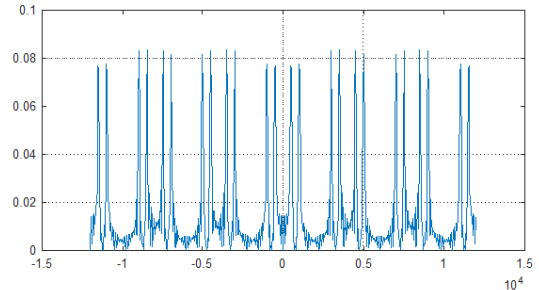
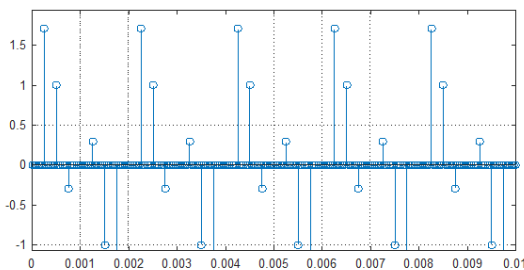
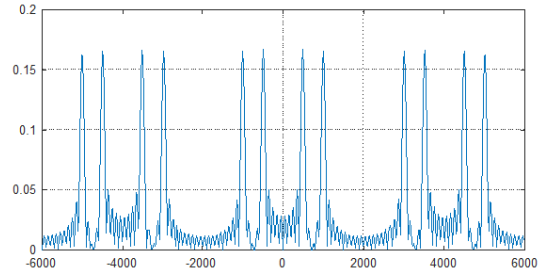
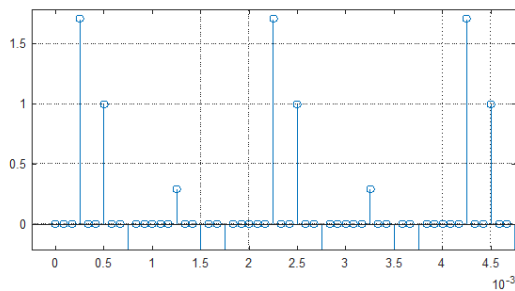
$$f_c = \frac{300}{2} = 150, f_c = \frac{300}{2} = 150$$

Decimation: Anti-aliasing

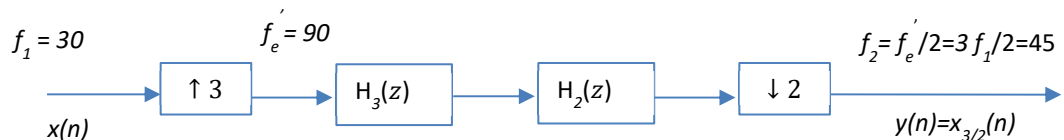
Interpolation: Anti-mirror (See course)

11. Gain Accuracy Before Digital-to-Analog Conversion

- $h(n) = \sin(2\pi \cdot 500 \cdot n) + \sin(2\pi \cdot 1000 \cdot n)$ $f_e = 4\text{kHz}$
- $D=3 \Rightarrow$ Insert 2 zeros between 2 consecutive samples. DTFT over 3 periods from -6kHz to 6kHz
- $D=6 \Rightarrow$ Insert 5 zeros between 2 consecutive samples. DTFT over 6 periods from -12kHz to 12kHz



12.



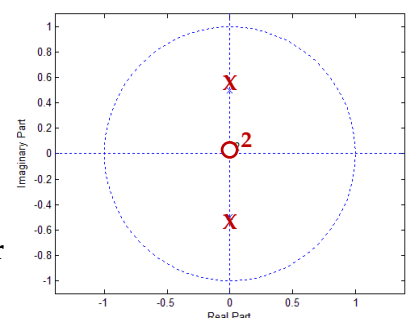
$$f_c = f_1/2 = f_e'/6 = 15$$

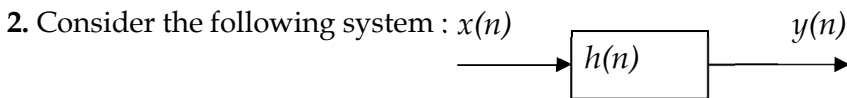
$$f_c = f_e'/4 = 22.5$$

Additional exercises

1. Suppose the plot of the poles and zeros of the following system:

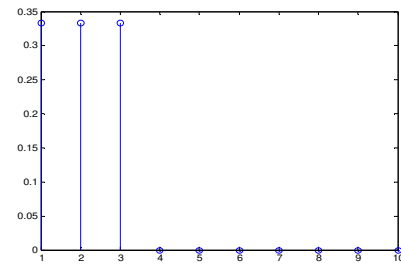
- Is it an FIR or IIR filter? (justify your answer)
- Give the approximate shape of $H(f)$
- Determine $H(z)$ then determine the recurrence equation
- Determine and plot $h(n)$
- From $h(n)$, study the stability, the causality and the invariance of the filter





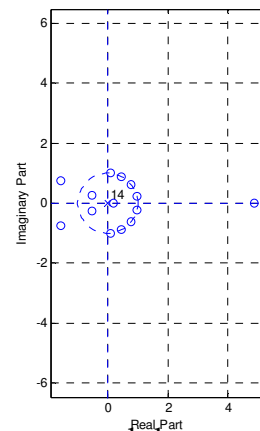
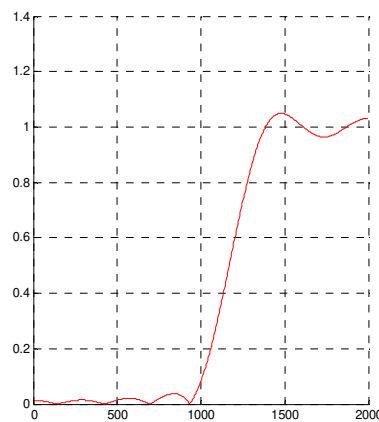
We assume that $h(n)$ is given as opposite.

- Study causality. Is it an FIR or IIR filter?
- From the expression of $h(n)$, deduce the role of this filter:
- Determine the recurrence equation of the system:
- Determine the poles and zeros of this filter then give their plot. Deduce an approximate plot of $|H(f)|$
- Calculate and plot $|H(f)|$ then deduce the plot of the modulus of the DFT for $N=6$.



3. During the transmission of a digital signal (sampled at a frequency of 2.5 kHz), it was affected by localized noise between the 350 Hz and 550 Hz frequency bands. We want to eliminate the noise by the use of an FIR filter having a transition band $\Delta f=100$ Hz. Design this filter using the windowing method. We want an attenuation in the attenuated band $A=-20 \log_{10}(\delta) > 20$ dB.

- Plot ideal $H(f)$
- Determine $h(n)$ and the order N of the filter
- Calculate $h(0)$, $h(1)=h(-1)$
- Then plot approximately $H(f)$



4. Plots of the modulus of the frequency response from 0 to $f_e/2$ and of the poles and zeros of a digital filter synthesized by the window method are given as follows:

- Is the phase shift of this filter linear? (Justify)
- Determine $H(k)$ then the impulse response $h(n)$
- Calculate $h(0)$.
- We want to use a hanning window. What happens to the plot of the frequency response modulus (superimpose it on $H(f)$)

5. We consider the third-order analog filter associated with the Butterworth approximation function. Realize the corresponding low-pass digital filter using the bilinear transformation. The cutoff frequency is $f_c=1$ kHz and the sampling frequency $f_e=10$ kHz.

6. We seek to achieve a digital filter equivalent to the denormalized analog filter of transmittance

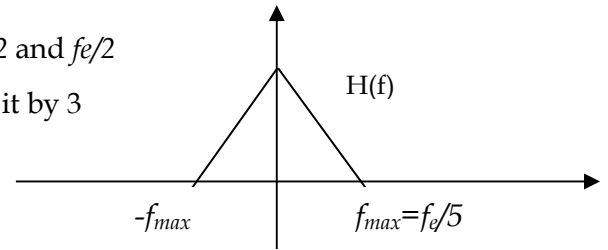
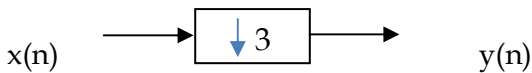
$$H_A(p) = \frac{1}{1 + 0.2p}$$

- Determine the impulse response $h_a(t)$
- Calculate the response in z of this filter obtained by bilinear transformation for a sampling frequency $T_e = 0.2$. What is its cutoff frequency? Calculate the z -response of a similar filter with the same cutoff frequency as the analog filter.

7. Consider the following signal: $h(n) = \sin(2\pi \cdot 150 \cdot n) + \sin(2\pi \cdot 350 \cdot n)$ and the modulus of the DFT whose respective plots are given above: (**see exo 9**)

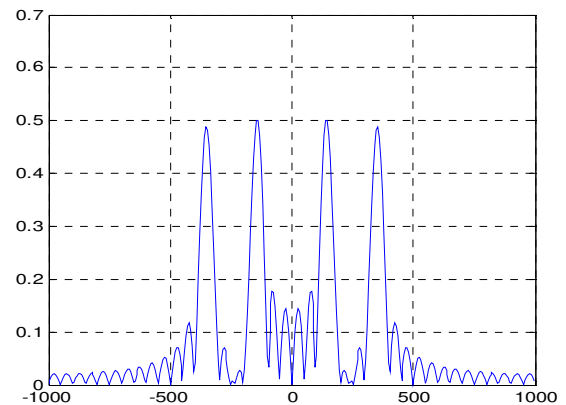
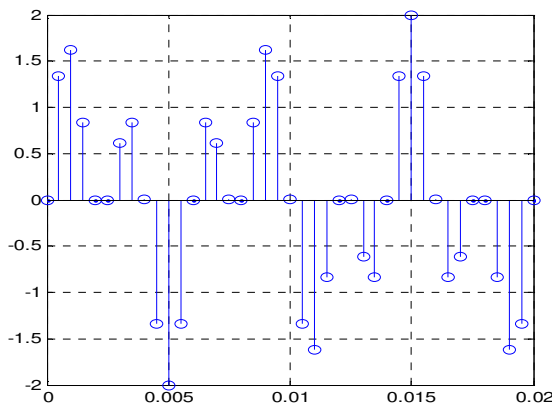
- Determine T_e .
- What is the point of interpolation?
- Plot the signal (the first 20 values) and the TF obtained for an interpolation of 2 then of 4 (without low-pass filtering)
- Why do we use a low-pass filter during interpolation, where do we place it?
- What type of Chebyshev filter is used in this case and why?
- Give the final polyphase decomposition for an interpolation of 2 by giving the expression of the polyphase filters

8. In order to transmit a signal $x(n)$ (whose DTFT between $-f_e/2$ and $f_e/2$ is given opposite) more quickly, we decide to decimate it by 3



- Plot the DTFT obtained after decimation ($f_e = 30\text{kHz}$)
- We want to place an anti-aliasing filter, where should we place it? (justify), plot the DTFT again
- Give the initial and final polyphase decomposition by giving the expression of each filter.

9. Consider the following signal: $h(n) = \sin(2\pi \cdot 150 \cdot n) + \sin(2\pi \cdot 350 \cdot n)$ and the modulus of the DFT whose respective plots are given below:



- Plot the signal and the TF obtained for a decimation of 2 then of 4 (with low-pass filtering)
- Give the final polyphase decomposition for a decimation of 4 by giving the expression of the polyphase filters
- What is the point of polyphase decomposition
- Why do we use a low pass filter during decimation?

Solutions: Interros and exams from previous years

Laboratory Work n°1: Synthesis of FIR and IIR filters

Purpose of the lab: In this lab, we test two different methods to synthesize a digital filter: an FIR filter by the window method and an IIR filter by the bilinear transformation method.

1. Reminders

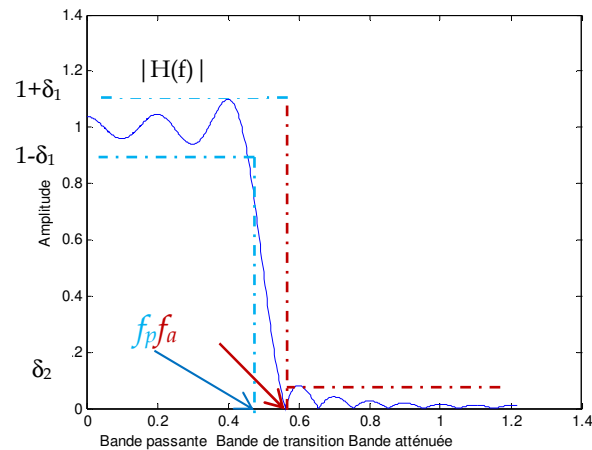
A finite impulse response (FIR) filter has a polynomial transfer function. It cannot be obtained by transposing a continuous filter, as is done for IIR filters. FIR filters have the disadvantage of requiring a large number of coefficients to obtain the same frequency characteristics. But on the other hand, they are unconditionally stable. It is possible to synthesize FIR filters with linear phase, that is to say with constant group propagation time.

The advantage of recursive filters (IIR) is their low computational cost. With very few poles and zeros we can provide most of the frequency responses that we may need in the different applications. The disadvantages of recursive filters are: their non-linearity in linear phase (linear phase: constant propagation time for any frequency), and their numerical instability. Indeed, with the filter being feedback-enabled, errors in numerical precision become a matter of importance, as they can amplify and become out of control, first in the form of noise, but eventually in the form of jitter. Note that IIR filters can be designed by methods similar to those used for analog filters.

Template of a filter

The template of a filter is none other than the set of characteristics of the filter, namely:

- The gain of the filter in the passband.
 - The attenuation of the notched band filter f_a .
 - The cutoff frequency f_c is often expressed in normalized form with respect to the sampling frequency.
 - The desired transition bandwidth Δf which must be as small as possible.
 - Any oscillations in bandwidth and/or attenuated.
- The determination of the coefficients of an FIR filter by the window method is carried out by the Python function `firwin` of the `scipy.signal` library.



2. Demo

-*- coding: utf-8 -*-

```
import numpy as np; import scipy. signal as sp; import matplotlib.pyplot as plt
from plot_zplane import zplane
#Design of a low-pass FIR filter fc=200, fe=1000, Deltaf=100, Aa>20
plt.figure(1)
fc=200;fe=1000; Deltaf=100
fcn=2*fc/fe; Deltafn=Deltaf/(fe/2)
Ncoef = int(np.ceil(1.8/Deltafn))
n = np.arange(-(Ncoef//2),Ncoef//2+1)
b = fcn*np.sinc(n*fcn);
#Visualization of impulse response and frequency response
L=256
f,Hf= sp.freqz(b,1,L,fs=fe);
plt.subplot(211); plt. stem(b);
plt.subplot(212); plt.plot(f, np.abs(Hf))
plt.title('Filter Module'); plt.xlabel('Frequency (Hz)');plt.ylabel('Amplitude')
plt.legend()
```

```

# Determination and plotting of poles and zeros
plt.figure(2)
a=np.array([1])
z,p = zplane(b,a)

# Effect of increasing the number of coefficients
fc=0.2;fe=1; fcn=2*fc/fe
A=np.array([21, 61, 101]);i=1;
a = np.array([1]);
L = 256;
plt.figure(3)
for N in A:
    n = np.arange(-(N//2),N//2+1)
    b = fcn*np.sinc(n*fcn);
    f,Hf= sp.freqz(b,a,L,fs=fe);
    plt.subplot(2,3,i); plt.stem(b);
    plt.subplot(212); plt.plot(f, np.abs(Hf),label='N=%d'%N)
    plt.title('Filter Module'); plt.grid(True); plt.xlabel('Frequency (Hz)');plt.ylabel('Amplitude')
    plt.legend()
    i+=1

#Effect of windowing
from scipy.signal import windows
plt.figure(4)
N=41;
A=np.array([np.ones(N), windows.hann(N),windows.hamming(N) ]);
i=1;
for Fen in A:
    n1 = np.arange(-(N//2),N//2+1)
    b1 = fcn*np.sinc(n1*fcn)*Fen;
    f,Hf1= sp.freqz(b1,a,L,fs=fe);
    plt.subplot(2,3,i); plt.stem(b1); plt.title('Filter * Fen %d'%i);
    plt.subplot(212); plt.plot(f, np.abs(Hf1),label='Win %d'%i)
    plt.title('Filter Module'); plt.grid(True); plt.xlabel('Frequency (Hz)'); plt.ylabel('Amplitude')
    plt.legend()
    i+=1

#Effect of windowing (display in db)
plt.figure(5)
fc=0.2;fe=2; fcn=2*fc/fe; N=41;
a = np.array([1]);
L=256;
A=np.array([np.ones(N), windows.hann(N),windows.hamming(N) ]);i=1;
for Fen in A:
    n2 = np.arange(-(N//2),N//2+1)
    b2= fcn*np.sinc(n2*fcn)*Fen;
    f,Hf1= sp.freqz(b2,a,L,fs=fe);
    plt.subplot(2,3,i); plt.stem(b2); plt.title('Filter * Fen %d'%i);
    plt.subplot(212); plt.plot(f, 20*np.log10(np.abs(Hf1)),label='Win %d'%i)
    plt.title('Filter Module'); plt.grid(True); plt.xlabel('Frequency (Hz)'); plt.ylabel('Amplitude')
    plt.legend()
    i+=1

# Bilinear transformation
fp=195;fa=205; fe=1000;
att_p=1; att_a=40;

```

```

wpm = fp*2*np.pi;
wam = fa*2*np.pi;
wpa = 2*fe*np.tan(np.pi*fp/fe) #Heartbeat compensation

N, WA = sp.buttord(wpm,wam, att_p,att_a, fs=fe)
z,p,k = sp.buttap(N); """Analog filter hn(p) in the form of poles, zeros and gain"""
# # sp.cheb2ap(N,att_a); sp.buttap(N); sp.ellipap(N,att_p,att_a)
Bpn,Apn = sp.zpk2tf(z,p,k); """Analog filter hn(p) in num, den form """
Bp, Ap = sp.lp2lp (Bpn,Apn,wpa); """Denormalization"""
# # sp.lp2hp sp.lp2bp sp.lp2bs
poles_anal = np.roots(Ap);
Bz1, Az1 = sp.bilinear(Bp,Ap,fe); """Transition from H(p) to H(z) by Bilinear Transformation"""

# Comparison of analog Ha(f) and digital Hz(f) filters
L = 256;
fz,HZ= sp.freqz(Bz1,Az1,L,fs=fe);
wa,Ha = sp.freqs(Bp,Ap, worN=np.arange(1,np.pi*fe)); fa=wa*0.5/np.pi;
plt.figure(6);
plt.plot(fz, np.abs(HZ),label='H(f) of the Digital filter');
plt.plot(fa, np.abs(Ha), label='H(f) of Analog filter'); plt.grid(); plt.legend()

plt.figure(7);
z,p=zplane(Bz1,Az1);

```

3. To do during lab work

```

# import numpy as np; import scipy. signal as sp; import matplotlib.pyplot as plt
# from plot_zplane import zplane
# import sounddevice as snd
# import scipy.io.wavfile as wav
# fe,x = wav.read('vousavezducourrierenattente.wav')
# snd. play(x, fe)

# Te=1/fe; N=len(x); t = np.arange(0,N)*Te;
# plt. figure(1); plt. subplot(211); plt. plot(t,x); plt.grid(True);
# plt.xlabel('time'); plt.ylabel('Amp'); plt.title('Phrase');

# # t = np.arange(0,N)*Te;
# # f0 = 1500; b = 20*np.cos(2*np.pi*f0*t);
# # Xb =x +b ;
# # snd.play(np.int8(Xb), fe)

```

"""

-----To do-----

Calculate and View your DFT in db in figure (1)

Add noise (3 sinusoids with a frequency of 1500, 2000 and 3000)

Listen to the noisy signal

View the DFT of the noisy signal in db

Design a filter by the window method with $A_a > 50$ and $\Delta f_a < f_e/10$, f_c to be determined using firwin

View Impulse Response and Filter Transfer Function

Filter the noisy signal X_b ($y = \text{sp.lfilter}(b,a,x)$) with the filter created (b,a)

Listen and view the DFT of the filtered signal and compare with the original'

Design Filter by the bilinear transformation method with $A_a > 50$ $A_p = 2$ with N to be determined ,
you have to try several analog filters (Butterworth, Cheby 1, Cheby 2), which one seems adequate to you?

View Impulse Response and Filter Transfer Function for each method

Filter the noisy signal X_b with the filter created

Listen and view the DFT of the filtered signal

Comment

Compare between the 2 filter synthesis methods. For this example, which do you think is preferable ?

''''''

Laboratory Work N°. 2: Multirate filtering

1. Interpolation and Decimation: Demo

```
# -*- coding: utf-8 -*-
import numpy as np; import matplotlib.pyplot as plt; import scipy.signal as sp;

N=30; f0=100; f1=300; fe=2000; Te=1/fe; NF=1024;
t = np.arange(0.0, N*Te, Te); x = np.sin(2.0*np.pi*f0*t) + np.sin(2.0*np.pi*f1*t);
TFx = np.fft.fft(x,NF); TFx = np.fft.fftshift(TFx); freq = np.arange(-NF/2,NF/2)*fe/NF;
plt.figure(1)
plt.subplot(321); plt.stem(t, x); plt.subplot(322); plt.plot(freq, np.abs(TFx));
x1 = np.zeros(2*N); x1[::2]=x[::]; t1 = np.arange(0.0, N*Te, Te/2); TFx1 = np.fft.fft(x1,NF);
TFx1 = np.fft.fftshift(TFx1); freq = np.arange(-NF/2,NF/2)*2*fe/NF;
plt.subplot(323); plt.stem(t1, x1); plt.subplot(324); plt.plot(freq, np.abs(TFx1));
x2 = np.zeros(4*N); x2[::4]=x[::]; t2 = np.arange(0.0, N*Te, Te/4); TFx2 = np.fft.fft(x2,NF);
TFx2 = np.fft.fftshift(TFx2); freq = np.arange(-NF/2,NF/2)*4*fe/NF;
plt.subplot(325); plt.stem(t2, x2); plt.subplot(326); plt.plot(freq, np.abs(TFx2));

plt.figure(2)
freq = np.arange(-NF/2,NF/2)*fe/NF;
plt.subplot(321); plt.stem(t, x); plt.subplot(322); plt.plot(freq, np.abs(TFx));
x1 = sp.resample(x,2*N); t1 = np.arange(0.0, N*Te, Te/2); TFx1 = np.fft.fft(x1,NF);
TFx1 = np.fft.fftshift(TFx1); freq = np.arange(-NF/2,NF/2)*2*fe/NF;
plt.subplot(323); plt.stem(t1, x1); plt.subplot(324); plt.plot(freq, np.abs(TFx1));
x2 = sp.resample(x,4*N); t2 = np.arange(0.0, N*Te, Te/4); TFx2 = np.fft.fft(x2,NF);
TFx2 = np.fft.fftshift(TFx2); freq = np.arange(-NF/2,NF/2)*4*fe/NF;
plt.subplot(325); plt.stem(t2, x2); plt.subplot(326); plt.plot(freq, np.abs(TFx2));

N=100; f0=100; f1=300; fe=2000; Te=1/fe; NF=1024;
t = np.arange(0.0, N*Te, Te); x = np.sin(2.0*np.pi*f0*t) + np.sin(2.0*np.pi*f1*t);
TFx = np.fft.fft(x,NF); TFx = np.fft.fftshift(TFx); freq = np.arange(-NF/2,NF/2)*fe/NF;
plt.figure(3)
plt.subplot(321); plt.stem(t, x); plt.subplot(322); plt.plot(freq, np.abs(TFx));
x1 = x[::2]; t1 = t[::2]; TFx1 = np.fft.fft(x1,NF); TFx1 = np.fft.fftshift(TFx1);
freq = np.arange(-NF/2,NF/2)*fe/(2*Nf);
plt.subplot(323); plt.stem(t1, x1); plt.subplot(324); plt.plot(freq, np.abs(TFx1));
x2 = x[::6]; t2 = t[::6]; TFx2 = np.fft.fft(x2,NF); TFx2 = np.fft.fftshift(TFx2);
freq = np.arange(-NF/2,NF/2)*fe/(6*Nf);
plt.subplot(325); plt.stem(t2, x2); plt.subplot(326); plt.plot(freq, np.abs(TFx2));

plt.figure(4)
freq = np.arange(-NF/2,NF/2)*fe/NF;
plt.subplot(321); plt.stem(t, x); plt.subplot(322); plt.plot(freq, np.abs(TFx));
x1 = sp.decimate(x,2); t1 = t[::2]; TFx1 = np.fft.fft(x1,NF); TFx1 = np.fft.fftshift(TFx1);
freq = np.arange(-NF/2,NF/2)*fe/(2*Nf);
plt.subplot(323); plt.stem(t1, x1); plt.subplot(324); plt.plot(freq, np.abs(TFx1));
x2 = sp.decimate(x,6); t2 = t[::6]; TFx2 = np.fft.fft(x2,NF); TFx2 = np.fft.fftshift(TFx2);
freq = np.arange(-NF/2,NF/2)*fe/(6*Nf);
plt.subplot(325); plt.stem(t2, x2); plt.subplot(326); plt.plot(freq, np.abs(TFx2));
```

2. Interpolation and Decimation: To do

```
# -*- coding: utf-8 -*-
```

```
import numpy as np; import matplotlib.pyplot as plt
#from zp_plot import zplane
import scipy.signal as sp;

from scipy.io import wavfile as wf; import winsound;
fname = 'youhavependingmail.wav';
winsound.PlaySound(fname, winsound.SND_FILENAME)
fe, x = wf.read(fname);
Te=1/fe; N=len(x); t = np.arange(0,N)*Te;
```

```
# plt.figure(1);plt.subplot(211); plt.plot(t,x); plt.grid(True);
# plt.xlabel('time'); plt.ylabel('Amp'); plt.title('Phrase');
# #Write to an audio file
# fnameb='Copy signal.wav';
# Xb=X
# Xb=np.int8(Xb); wf.write(fnameb, fe, Xb);
# winsound.PlaySound(fnameb,winsound.SND_FILENAME);

#-----To do-----
#1
# Test downsampling with and without low-pass filtering
# Display the signal and its TFD in each case and zoom in on the same area for a duration of 50 ms and
comment
# Listen to the obtained sign each time
# Is there any distortion of the sound? What is it due to?
# Up to what decimation factor can we go without deformation
# What happens if we keep the same (original) fe playing?

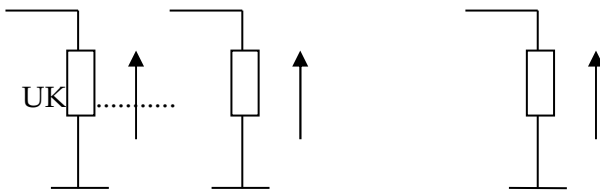
#2
# Test oversampling with and without lowpass filtering
# Display the signal and its TFD in each case and zoom in on the same area for a duration of 50 ms or 20
ms and comment
# Listen to the signal obtained each time (wf.wite with the new fe by multiplying by the Interp factor)
# Do we perceive a distortion of the sound? Why?
# What happens if you keep listening to the same fe?

#3
# We want to move to a signal whose sampling frequency is  $1.5 \cdot f_e$ 
# Propose 2 solutions
# Which one allows you to preserve the original values the most?
```

II. Random processes

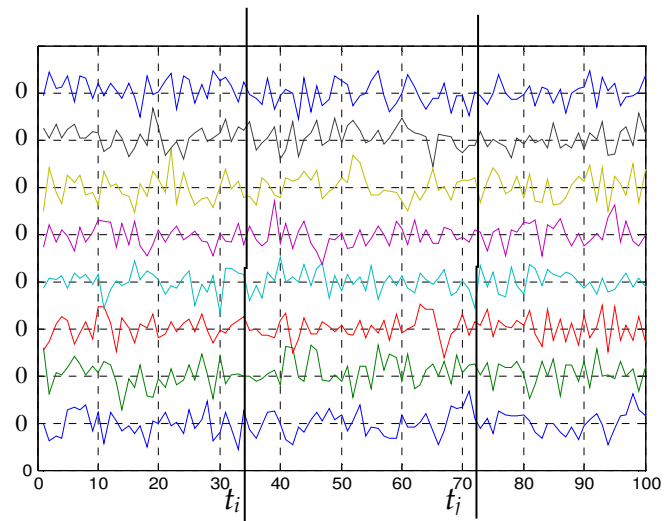
Random (stochastic) processes describe the evolution of a random quantity as a function of time (or space). We can define a stochastic process as being a family of random variables indexed by time defined on the same space of probabilities Ω . A random process does not have analytic temporal representations. Each observed random signal represents a particular realization of this process. The speech signal, the radar signal, the electrocardiogram, the electro-encephalogram, the seismic signals are examples. Recall that a reception signal - consisting of an informative signal (random or deterministic), a random interference signal and noise related to the transmission channel - is random by nature.

Example 1: If we take several identical resistances (of the same value) and we measure the voltage. One will find a non-zero value due to the thermal agitation of the free electrons in the resistance. The voltages measured over time on all of the resistors will provide several random signals, all different, which constitute the random process.



Signals that can be produced by measuring the voltage across multiple resistors.

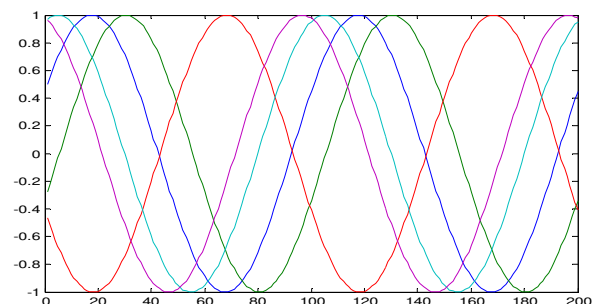
- Each plot provides a random signal
- at time t_i , the process is reduced to a value x_i whose density is $p(x; t_i) = p(x_i)$



- Two instants t_i and t_j make it possible to define two random variables x_i and x_j , we can define joint probability densities by: $p(x; t_i, t_j) = p(x_i, x_j)$

Example 2: Random phase sinusoidal signal $X(t, \varphi) = a \cos(\omega t + \varphi)$ with phase φ uniformly distributed between 0 and 2π

- φ is a continuous real value
- $X(t, \varphi)$ has a continuous and real value
- A particular signal $X(t, \varphi_i)$ is deterministic



In practice, it is not easy to obtain the probability density $p(x_i)$ moments of order 1 and 2

we are then satisfied with

1. Mathematical Expectations

1st order statistics : They allow the signal to be described at a given instant. The random process becomes a simple random variable that can be described using moments provided that its probability is known $\forall t_i$.

- statistical average : $\mu_x(t_i) = E\{x(t_i)\} = E\{x_i\} = \int_{-\infty}^{+\infty} x(t_i) \cdot p(x, t_i) dx$

- Variance : $\sigma_x^2(t_i) = E\{x(t_i) - \mu_x(t_i)\}^2 = E\{x_i^2\} - \mu_x(t_i)^2$

Where $E\{x_i^2\} = \int_{-\infty}^{+\infty} x(t_i)^2 \cdot p(x, t_i) dx$ is the mean square value (we also speak of power) and $\sigma_x(t_i)$ is said standard deviation

Example 1 : Let the stochastic process $x(t)$ be defined by: $x(t) = a + bt$ where a and b are r.v.s whose probabilities are known, then:

$$\mu_x(t) = E[a + bt_i] = \mu_a + \mu_b t$$

Let us return to example 2 : For a given time t_k , we can calculate statistical moments of the random variable $X(t_k, u)$

- Average :

$$\mu_x(t) = E[X(t)] = \int_0^{2\pi} f_\varphi(\varphi) a \cos(\omega t + \varphi) d\varphi = \frac{1}{2\pi} a \int_0^{2\pi} \cos(\omega t + \varphi) d\varphi = 0$$

-Variance:

$$\sigma_x^2(t) = E[X^2(t)] - \mu_x^2(t) = E[X^2(t)] = \int_0^{2\pi} f_\varphi(\varphi) a^2 \cos^2(\omega t + \varphi) d\varphi = \frac{a^2}{2}$$

Suppose now that φ is deterministic and that a is random with density $p(a)$, mean μ_a and variance σ_a^2 , then:

$$\mu_x(t) = E[a \cos(\omega t + \varphi)] = \mu_a \cdot \cos(\omega t + \varphi) \quad \sigma_x^2(t) = \sigma_a^2 \cdot \cos^2(\omega t + \varphi)$$

2nd order statistics :

We say that the process is known at 2 times t_1 and t_2 , if $\forall t_1, t_2$, the joint probability is known. Insofar as the expectation involves the product of two random variables (2 instants), we will speak of statistics of order 2. we can estimate the following statistics:

- Statistical autocorrelation function :

$$R_x(t_1, t_2) = E\{x(t_1) \cdot x^*(t_2)\} = E\{x_1 \cdot x_2^*\} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x_1 \cdot x_2^* \cdot p(x_1, x_2) \cdot dx_1 \cdot dx_2$$

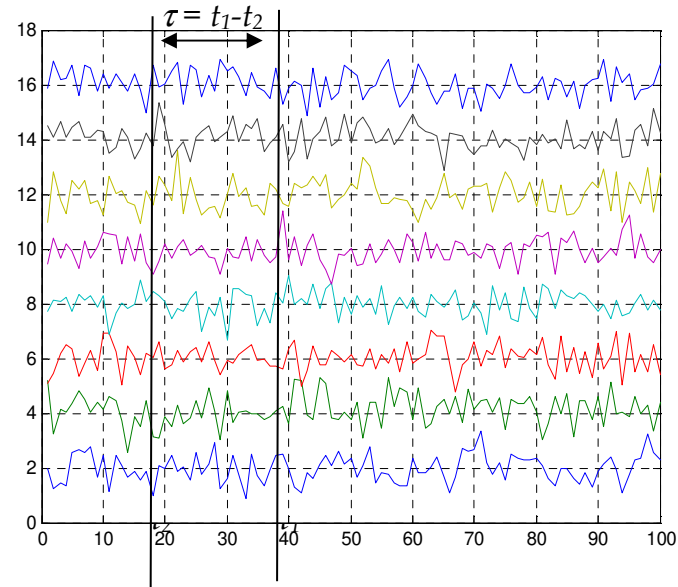
- Statistical autocovariance function:

$$C_x(t_1, t_2) = E\{[x(t_1) - \mu_x(t_1)] \cdot [x(t_2) - \mu_x(t_2)]^*\} = R_x(t_1, t_2) - \mu_x(t_1) \cdot \mu_x(t_2)^*$$

They illustrate the relationship between the statistics taken at **two** different times t_1 and t_2 . The *autocorrelation* function measures the correlation between the signals emitted by the *same* process at two *different instants*. When there is correlation, we can speak of a "memory effect" of the process. We define the memory of the process as the time t_c beyond which the correlation is negligible, t_c is also called correlation time. On the other hand, if it is zero, the signal is completely random and the signal $x(t)$ at time t is completely decorrelated from this same signal at time past $t - \tau$.

centered processes

$$(\mu_x(t)=0) : C_x(t_1, t_2) = R_x(t_1, t_2)$$



We can extend these notions to measure this link between two random processes by:

- Statistical correlation function: $R_{xy}(t_1, t_2) = E\{x(t_1) \cdot y^*(t_2)\} = E\{x_1 \cdot y_2^*\}$
- Statistical covariance function : $C_{xy}(t_1, t_2) = R_{xy}(t_1, t_2) - \mu_x(t_1) \cdot \mu_y^*(t_2)$
- The correlation coefficient is defined by: $\rho_{xy}(t_1, t_2) = \frac{C_{xy}(t_1, t_2)}{\sigma_x(t_1)\sigma_y(t_2)} \quad -1 \leq \rho_{xy}(t_1, t_2) \leq 1$

Let's go back to example 2 and calculate the statistical auto-correlation:

$$R_{xx}(t_1, t_2) = E[a \cos(\omega t_1 + \varphi) a \cos(\omega t_2 + \varphi)] = E\left[\frac{a^2}{2} (\cos(\omega(t_1 + t_2) + 2\varphi) + \cos(\omega(t_1 - t_2)))\right] = \frac{a^2}{2} \cos(\omega(t_1 - t_2))$$

Note that it is a periodic function, depending only on the difference $t_1 - t_2$. For $t_1 = t_2$, we find the variance of the random process, ie $a^2/2$.

3. Stationary processes

Many random processes observed in practice have statistical properties that do not depend on the time the observation is made. They are said to be stationary processes.

We designate by *stationary* the processes whose *statistical characteristics* are *independent of the origin of time*. A random process is said to be stationary in the *strict sense* when all these statistical characteristics, ie all its moments *at any order*, are independent of the origin of time.

$$\begin{aligned} \Leftrightarrow p(x, t_i) &= p(x); \\ \mu_x(t_i) &= \mu_x \\ R_{xy}(t_1, t_2) &= R_{xy}(\tau) \text{ avec } \tau = t_2 - t_1 \\ &\dots \end{aligned}$$

Stationarity does not mean that the process is independent of time, but rather that its statistical properties do not depend on the moment at which one begins to estimate them. Thus, temperature is a non-stationary process while dice rolling is a stationary process.

Many phenomena can be considered to be approximately stationary over finite observation times. As we do not always have access to $p(x, t_i)$, we will be satisfied with *stationarity in the broad sense* (SSL) when only μ_x and R_x are independent of t .

So :

- A stationary process is said to be stationary of order 1 if its mean and its variance are constant and therefore independent of any time lag:

And $\mu_x(t) = \mu_x = \text{cste} \quad \sigma_x^2(t_i) = \sigma_x^2$

- A stationary process is said to be stationary of order 2 if its statistics of order 2 depend only on the time difference between the two instants t_1 and t_2 :

$$R_x(t_1, t_2) = R_x(\tau) \text{ avec } \tau = t_2 - t_1$$

Let us return to the previous example $X(t, u) = a \cos(\omega t + u)$ and study its stationarity in the broad sense when u is uniform on $[0, 2\pi]$: the mean, the variance are independent of t and the statistical autocorrelation only depends on $t_1 - t_2$. The process is therefore stationary in the broad sense.

Application example:

Show that the process $x(t) = a \cdot \cos(2\pi t) + b \cdot \sin(2\pi t)$ where a, b are decorrelated variables with zero mean and unit variance is stationary in the broad sense.

Properties of the autocorrelation function for SSL real random x

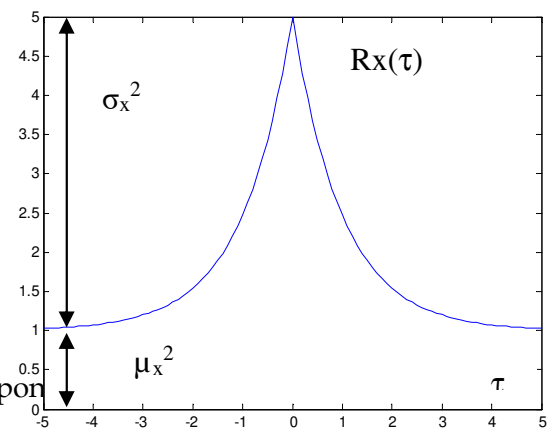
-The correlation function is even: $R_x(\tau) = R_x(-\tau)$ avec $\tau = t_2 - t_1$

- The function is maximum at the origin $R_x(0) \geq |R_x(\tau)|$

- Values at the origin and at infinity: $R_x(0) = E(x(t)^2) = \mu_x^2 + \sigma_x^2$

And $\lim_{\tau \rightarrow \infty} R_x(\tau) = \lim_{\tau \rightarrow \infty} E\{x(t)x(t-\tau)\} = \mu_x^2$

Indeed, if $x(t)$ contains neither periodic nor time-independent component and $x(t - \tau)$ become statistically uncorrelated.



Discrete case:

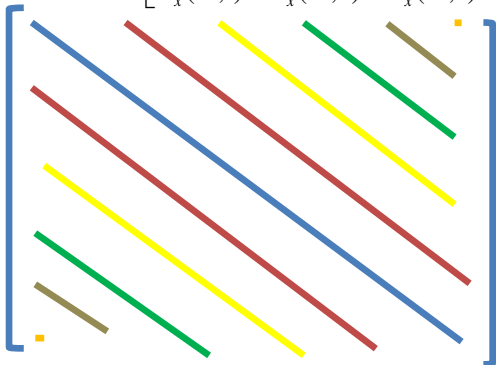
We will replace t by $n \rightarrow \mu_x(n) = \mu_x = E\{x(n)\} \quad R_x(n_1, n_2) = E\{x(n_1)x^*(n_2)\}$

A random process X discrete SSL will have a constant mean and a statistical autocorrelation depending only on $k = n_1 - n_2$ being:

$$\mu_x(n) = \mu_x = \text{cste} \quad R_x(n_1, n_2) = R_x(k) \text{ avec } k = n_1 - n_2$$

$$R_x(n_1, n_2) = \begin{bmatrix} R_x(0) & R_x(1) & R_x(2) & \dots & R_x(N-1) \\ R_x(1) & R_x(0) & R_x(1) & \dots & R_x(N-2) \\ R_x(2) & R_x(1) & R_x(0) & \dots & R_x(N-3) \\ \dots & \dots & \dots & \dots & \dots \\ R_x(N-1) & R_x(N-2) & R_x(N-3) & \dots & R_x(0) \end{bmatrix}$$

SSL $\Rightarrow R_x(n_1, n_2) = R_x(n_1 - n_2) = \begin{bmatrix} R_x(1,1) & R_x(1,2) & R_x(1,3) & \dots & R_x(1,N) \\ R_x(2,1) & R_x(2,2) & R_x(2,3) & \dots & R_x(2,N) \\ R_x(3,1) & R_x(3,2) & R_x(3,3) & \dots & R_x(3,N) \\ \dots & \dots & \dots & \dots & \dots \\ R_x(N,1) & R_x(N,2) & R_x(N,3) & \dots & R_x(N,N) \end{bmatrix}$



$$R_x(n_1 - n_2) = R_x(k)$$

Thus, the correlation matrix of a discrete stationary process is a square Toeplitz matrix. A square matrix is called Toeplitz if all the elements of the same diagonal or subdiagonal are equal. We can directly see that this is the case here. This property is directly linked to the property of stationarity (in the broad sense) of the process.

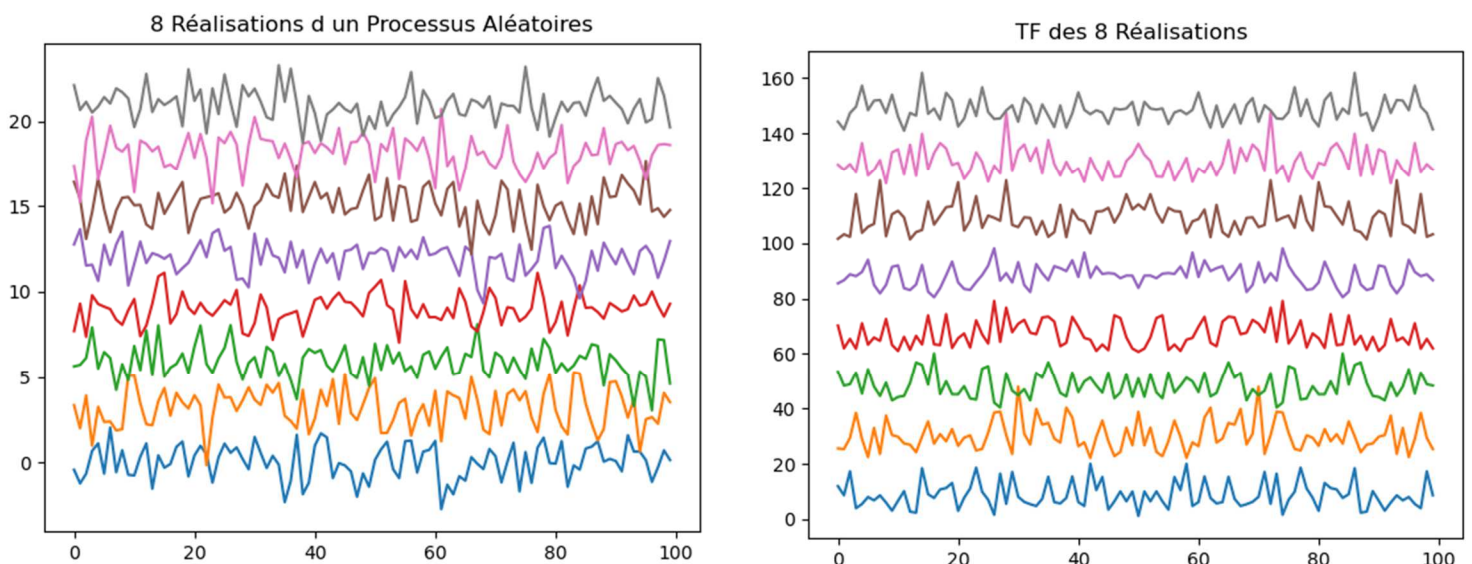
Power and DSP: People who are conversing (in a café or in class) generate a random signal which, depending on its overall volume, will have a power. If $X(t)$ is SSL, we can calculate the expectation of the instantaneous power by:

$$P_X = E(x(t)^2) = R_x(0) = \mu_x^2 + \sigma_x^2$$

The energy is then given by:

$$E_X = \int P_X dt = \int E(x(t)^2) dt$$

As for the spectrum for a random signal, it should be noted that each realization will provide a different spectrum (See example below)



However, we already have a statistical tool that contains unique information when the process is considered SSL: it is the statistical correlation function, whose TF will provide information on the frequency distribution of the signal's mean power.

We then define the power spectral density of a broadly stationary random signal $X(t)$ as the function of the frequency f given by the TF of the statistical correlation function of the signal:

$$S_x(f) = \int_{-\infty}^{+\infty} R_x(\tau) e^{-j2\pi f\tau} d\tau$$

The total average power of the process is: $P_x = \int_{-\infty}^{+\infty} S_x(f) df = R_x(0)$

As the spectrum represents an average over the set of possible realizations of the process, a particular realization can always have a power spectrum different from $S_x(f)$.

White noise: White noise is a second-order stationary random signal whose power spectral density is constant along the entire frequency axis.

It is defined by: $R_x(\tau) = \sigma_x^2 \delta(\tau)$ which implies that its mean will be zero ($\mu_x = 0$) and that it is decorrelated.

The TF of the statistical correlation is then given by: $S_x(f) = \sigma_x^2$

In the discrete case, we will have $R_x(k) = \sigma_x^2 \delta(k) \Rightarrow$ and $R_x(k) = 0$ pour $k \neq 0$ $\mu_x(k) = 0$

We then say that the noise is uncorrelated.

By analogy with white light, we call such a signal white noise. There are therefore different types of white noise depending on the random variable describing the noise. The most common are Gaussian white noise where the random variable follows a Gaussian law and uniform white noise where the random variable follows a uniform law.

4. Ergodic processes

It is not always possible to perform a sufficient number of measurements to establish the statistical properties of a random process. It is easier to roll a die 1000 times than to requisition 1000 people to roll 1000 dice. Just as for characterizing thermal noise, several *measurements* would be taken using the same resistor instead of using 1000 . achievements.

A stationary random process is said to be *ergodic* when the statistical and temporal mean values are identical .

$$\mu_x = \overline{\mu_x}; \quad R_x(\tau) = \overline{R_x(\tau)} \dots$$

$$\text{where } \overline{x} = \lim_{T \rightarrow +\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) \cdot dt \quad \text{and} \quad \overline{R_{xy}(\tau)} = \overline{x(t) \cdot y^*(t-\tau)}$$

From a practical point of view, when a random phenomenon is ergodic and stationary, it can be measured with a single reliable device from any instant.

$$\overline{\mu_x} = \lim_{T \rightarrow +\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) \cdot dt \Rightarrow \mu_x = \overline{\mu_x} \quad \overline{R_x(\tau)} = \lim_{T \rightarrow +\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t)x(t-\tau) \cdot dt \Rightarrow R_x(\tau) = \overline{R_x(\tau)}$$

Let's go back to example 2 for the umpteenth time and calculate the temporal averages:

$$E[X(t, u_i)] = \overline{X}(u_i) = \frac{1}{T} \int_{-T/2}^{+T/2} a \cos(\omega t + u_i) dt = 0 \quad E[X^2(t, u_i)] = \overline{X^2}(u_i) = \frac{1}{T} \int_{-T/2}^{+T/2} a^2 \cos^2(\omega t + u_i) dt = \frac{a^2}{2}$$

It is therefore ergodic of order 1

If the process is ergodic $S_x(f)$ represents the power spectrum of any realization $x_i(t)$:

$$P_x = \int_{-\infty}^{+\infty} S_x(f) df = \lim_{T \rightarrow +\infty} \frac{1}{T} \int_{-T/2}^{T/2} x_i(t)^2 dt \quad \text{Where } x_i(t) \text{ is a realization of } x(t).$$

In the case where the signal $x(t)$ is ergodic, stationary, we determine $S_x(f)$ by calculating the temporal DSP on T :

$$S_x(f) = TF\{R_x(\tau)\} = \lim_{T \rightarrow \infty} \frac{1}{T} |X_T(f)|^2 \quad \text{Where } X_T(f) \text{ is the TF of } x(t) \text{ limited to the interval } [0, T]$$

5. Process Examples

There are many classes of particular processes: Markov processes (and in particular Markov chains when t is discrete), martingales, Gaussian processes, Poisson processes.

- **Poisson**

The Poisson distribution is intended to model the frequency of events during a fixed time interval. Given a time interval of length, t , and the event arrival rate, λ , the expected number of events during this time interval is:

$$\mu = \lambda \cdot t$$

For the Poisson process to be feasible, the following assumptions must be fulfilled :

1. The probability of success over a very short period of time is the λ parameter multiplied by this period of time.
2. The probability of more than one success event occurring within the defined time interval is not significant. In other words, the probability of more than one experiment succeeding in a fixed time interval is very small, and therefore not large or not significant.
3. The probability of a successful event occurring during a defined time interval does not depend on what happened previously. In other words, each successful experiment is independent of the previous experiment.

The Poisson process is known in statistics as a stochastic process that attempts to record very improbable events in continuous time.

Examples: breakdown, nuclear accident, manufacturing defect in a chain. It is also used in Telecommunications to estimate the number of calls in a given period of time.

- *Markovian*

Markov's axiom translates that the probability of any future behavior, the present being known, is not modified by any additional knowledge of the past.

A process is Markovian, if for any instant u , for any given value $X_u = x$, the probability that the process takes the value y at any subsequent instant t ($t > u$), does not depend on the values taken by the process before time u . The process is said to be memoryless.

Markov chains are Markovian processes that evolve in discrete time according to probabilistic transitions. They will therefore describe a set of random phenomena according to the principle of stochastic processes.

Examples: the height of the level of a dam at a time t , the price of oil at the end of the day

- *Gaussian*

Gaussian processes play a crucial role in stochastic process theory because:

- Some stochastic processes can be approximated by Gaussian processes,
- Calculations are facilitated within the framework of Gaussian processes.
- Central limit theorem and approximations: Whenever a phenomenon can be considered as the result of a large number of independent random causes (e.g. exam marks, child's weight), one can summarize that this phenomenon follows a law of normal distribution: this is the central limit theorem. Thus, the statistical distribution of the sum of n independent AVs with the same distribution tends towards the Normal distribution when n tends towards infinity.

A real process (X_t) is Gaussian if: $\forall n \forall t_1; \dots; t_n$, the vector $(X_{t_1} = x_1; \dots; X_{t_n} = x_n)$ is Gaussian.

$$Y = \sum_{k=1}^n \alpha_k X_k = \alpha^T X$$

Recall that all the marginal probabilities of a Gaussian process are Gaussian and that any linear combination of marginals of a Gaussian process is Gaussian.

$$p(x_1, x_2, \dots, x_n) = \frac{1}{\sqrt{(2\pi)^n \det(C_X)}} e^{-\frac{1}{2}(\underline{x} - \underline{\mu}_X)^T C_X^{-1} (\underline{x} - \underline{\mu}_X)}$$

Or

$$\underline{X} = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix}, \underline{\mu}_X = \begin{bmatrix} \mu_{X1} \\ \vdots \\ \mu_{Xn} \end{bmatrix}, \underline{X}_C = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix} - \begin{bmatrix} \mu_{X1} \\ \vdots \\ \mu_{Xn} \end{bmatrix}$$

Example: A real (standard) Brownian motion is a centered Gaussian process with continuous trajectories. It is a process with independent stationary and Gaussian increases.

6. Estimation of spectral density

The quantities $E\{x\}$, $E\{x^2\}$, $S_{xx}(f)$,... are impossible to calculate on a computer because they would require an infinite number of points. On a calculator, we only have a discrete and finite sequence of N points. In reality, we calculate estimates of these quantities by generally assuming the signal is stationary and ergodic. We then replace the calculation of statistical averages with temporal averages.

We will calculate the DSP on a number of finite points. Thus, the Fourier transform calculated on the finite sequence will therefore be convolved with the spectrum of the rectangular window, that is to say a cardinal sine. Let us recall that the spectral properties of the cardinal sine are not well suited to the spectral analysis of the signal (significant secondary lobes). To remedy this, we use weighting windows (Hamming, Hanning, Kaiser, etc.)

The periodogram method to take the TF of an achievement: $\hat{S}_{xx}(f) = \frac{1}{N} |Y(f)|^2$ avec $Y(f) = \sum_{k=0}^{N-1} y_k e^{-2\pi jfk}$

$$\hat{S}_{xx}(f) = \frac{1}{N} \sum_{l=0}^{N-1} y_l e^{-2\pi jfl} \sum_{m=0}^{N-1} y_m e^{2\pi jfm} = \frac{1}{N} \sum_{l=0}^{N-1} \sum_{m=0}^{N-1} y_l y_m e^{-2\pi jf(l-m)}$$

$$\text{From where} \quad = \frac{1}{N} \sum_{k=0}^{N-1} \sum_{l=k}^{N-1} y_l y_{l-k} e^{-2\pi jfk} = \sum_{k=0}^{N-1} \hat{R}_{yy}(k) e^{-2\pi jfk} = TF(\hat{R}_{yy}(k))$$

Calculating the expectation gives $\hat{S}_{xx}(f)$:

$$E\{\hat{S}_{xx}(f)\} = E\left\{ \sum_{k=0}^{N-1} \hat{R}_{yy}(k) e^{-2\pi jfk} \right\} = E\left\{ \frac{1}{N} \sum_{k=0}^{N-1} \sum_{l=k}^{N-1} y_l y_{l-k} e^{-2\pi jfk} \right\}$$

$$E\{\hat{S}_{xx}(f)\} = \frac{1}{N} \sum_{k=0}^{N-1} \sum_{l=k}^{N-1} E\{y_l y_{l-k}\} e^{-2\pi jfk} = \sum_{k=0}^{N-1} \frac{N-k}{N} R_{yy}(k) e^{-2\pi jfk} = S_{yy}(f) * N \text{sinc}(fN)$$

The periodogram is a biased estimator. The periodogram is therefore on average the convolution of the true spectrum with the Fourier transform of the triangular window. Nevertheless, when $N \rightarrow \infty$, the bias becomes zero. The variance is practically independent of N and proportional to the spectrum: $\propto S_x(f)^2$. Note that the quality of the estimation by periodogram will be all the better as the signal is observed over a long range of stationarity.

In order to reduce the variance of this estimator, we can use an averaged periodogram. This consists of separating the signal into K slices (of length N/K), calculating the periodogram on each slice and taking the average. Due to the K averaging, the variance is almost divided by K ; however, the slices being shorter, the resolution decreases ($\Delta f = f_e/N \rightarrow f_e/K$). In practice, a recovery rate of 40% gives good results. Beyond that, the independence between the sections is no longer respected.

b. The correlogram method consists of first calculating the estimate of $R_{xx}(k)$ of the autocorrelation function then taking the TF of this estimate as an estimate of the spectral density.

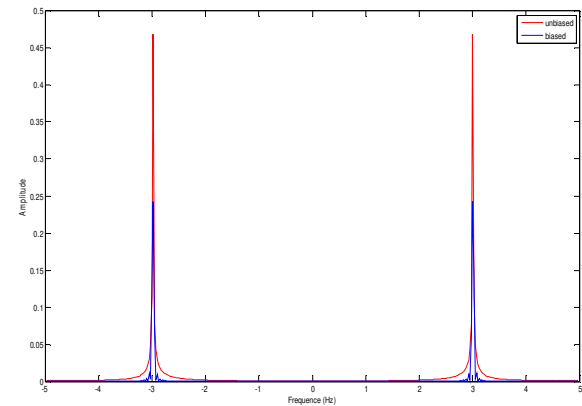
$$\hat{S}_{xx}(f) = TF[\hat{R}_{xx}(k)]$$

From a sample $(x_0, x_1, \dots, x_{N-1})$ independent and identically distributed, several estimators are then possible:

$$\hat{R}_{xx}(k) = \frac{1}{N} \sum_{n=k}^{N-1} x(n)x^*(n-k) \quad \hat{R}_{xx}(k) = \frac{1}{N-k} \sum_{n=k}^{N-1} x(n)x^*(n-k)$$

The first estimator is biased because the number of available samples which varies with step k is not taken into account. The second takes this into account, it is unbiased.

The correlograms each have an extrema for the normalized frequency 3 which is indeed the frequency of the sinusoid studied. It also emerges from these curves that the unbiased correlogram makes it possible to better highlight the frequency of the sinusoid but at the expense of a greater variance on the edges.



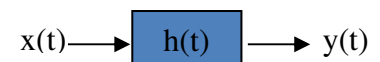
c) In this method, we search for an AR, MA or ARMA model for the sequence x_k . In the case of an autoregressive model $x_k = a_1 x_{k-1} + \dots + a_r x_{k-r} + u_k$

$$\hat{S}_{xx}(f) = \left| \frac{1}{1 + \sum_{k=1}^r a_k e^{-i2\pi f k}} \right|^2 u_0$$

7. Filtering Random Processes

A random signal is caused to be transmitted, analyzed, transformed, etc. Does it retain its randomness? its stationarity? What happens to its statistical mean and autocorrelation during linear filtering. We will examine the transformation of the signal characteristics in the frequency domain which will make it possible to approach the notion of shaping filter. This knowledge will allow us to consider a direct application which is optimal and Matching filtering.

Consider a linear and time-invariant system (LTI) defined by its impulse response $h(t)$ or its transfer function $H(f)$:



Recall that the response of the linear and invariant system to any deterministic signal is given by:

$$y(t) = x(t) * h(t) = \int x(\tau)h(t-\tau)d\tau$$

This expression implies that if $x(t)$ is a random signal, the output signal $y(t)$ is necessarily a random signal since the output is a weighted sum of the input. It will therefore be necessary to characterize $y(t)$ statistically, as well as for $x(t)$, by studying the statistical quantities already seen in the previous chapter.

Statistical mean and autocorrelation of $y(t)$

Suppose that $x(t)$ is a random process, the average $\mu_y(t)$ of the output signal (also random) can be calculated by:

$$\begin{aligned} \mu_y(t) &= E\{y(t)\} = E\{x(t) * h(t)\} = E\left\{\int h(\tau)x(t-\tau)d\tau\right\} \\ &= \int h(\tau)E\{x(t-\tau)\}d\tau = \int h(\tau)\mu_x(t-\tau)d\tau = h(t) * \mu_x(t) \end{aligned}$$

In the same way, we can calculate the statistical auto-correlation of $y(t)$ that is $R_y(t_1, t_2)$:

$$\begin{aligned} R_y(t_1, t_2) &= E\{y(t_1)y^*(t_2)\} = E\{x(t_1) * h(t_1).x^*(t_2) * h^*(t_2)\} \\ &= E\left\{\int h(\tau_1)x(t_1 - \tau_1)d\tau_1 \int h^*(\tau_2)x^*(t_2 - \tau_2)d\tau_2\right\} \\ &= \int \int h(\tau_1)h^*(\tau_2)E\{x(t_1 - \tau_1)x^*(t_2 - \tau_2)\}d\tau_1d\tau_2 \end{aligned}$$

If $x(t)$ is broadly stationary (SSL) then:

- The statistical average of $y(t)$ becomes:

$$\mu_y(t) = \int h(\tau)\mu_x(t - \tau)d\tau = \int h(\tau)\mu_x d\tau = \mu_x \int h(\tau)d\tau = \mu_x H(0) \quad (f = 0)$$

- The statistical correlation of y becomes:

$$\begin{aligned} R_y(t_1, t_2) &= \int \int h(\tau_1)h^*(\tau_2)E\{x(t_1 - \tau_1)x^*(t_2 - \tau_2)\}d\tau_1d\tau_2 \\ &= \int \int h(\tau_1)h^*(\tau_2)R_x(\tau - \tau_1 + \tau_2)d\tau_1d\tau_2 = fct(\tau) = R_y(\tau) \end{aligned}$$

So if the input signal is stationary in the broad sense (SSL) then the output signal will also be SSL

$$\text{Knowing that } f(\tau) * h(\tau) = \int f(\tau_2)h(\tau - \tau_2)d\tau_2 = \int f(\tau - \tau_2)h(\tau_2)d\tau_2$$

$$\text{SO } f(\tau) * h(-\tau) = \int f(\tau_2)h(\tau_2 + \tau)d\tau_2 = \int f(\tau + \tau_2)h(\tau_2)d\tau_2$$

$$\text{Therefore } R_y(\tau) = R_x(\tau) * h(\tau) * h^*(-\tau)$$

Spectral density

By applying the Fourier transform to both sides of the previous equation, the power spectral density is obtained:

$$S_y(f) = TF(R_y(\tau)) = TF(R_x(\tau) * h(\tau) * h^*(-\tau)) = S_x(f).H(f).H^*(f) = |H(f)|^2 S_x(f)$$

Thus, the power spectral density of the output signal is equal to the power spectral density of the input signal multiplied by the square of the modulus of the frequency response of the system (the phase of $H(f)$ does not intervene). It is a very important property at the base of many applications including in particular the notion of forming filter and optimal filtering.

Moreover, if it is desired to calculate the correlation of the output signal by avoiding the heaviness of calculation inherent in the convolution product, the power spectral density of the output signal will be calculated and the inverse Fourier transform will be taken.

$$R_y(\tau) = TF^{-1}(|H(f)|^2 S_x(f)) = \int |H(f)|^2 S_x(f) e^{2\pi j f \tau} df$$

The average power of the output signal is then obtained by: $E\{y(t)^2\} = R_y(0) = \int |H(f)|^2 S_x(f) df$

Interference Formula

Let $y_1(t)$ be the output of a system $h_1(t)$ whose input is a random signal $x_1(t)$ and $y_2(t)$ the output of a system $h_2(t)$ whose input is a random signal $x_2(t)$. The formula of the interference makes it possible to relate the cross-correlation between the outputs of two filters to those of the inputs of these filters:

$$S_{y_1 y_2}(f) = H_1(f) S_{x_1 x_2}(f) H_2^*(f)$$

We also have : $R_{xy}(\tau) = R_x(\tau) * h^*(-\tau) \Rightarrow S_{xy}(f) = S_x(f) H^*(f)$

And $R_{yx}(\tau) = R_x(\tau) * h(\tau) \Rightarrow S_{yx}(f) = H(f) S_x(f)$

Example 1: Consider the stochastic process $x(t)$ SSL of statistical correlation: $R_x(\tau) = \sigma^2 e^{-\frac{|\tau|}{\theta}}$ and the linear and invariant impulse response system $h(t) = 5e^{-2t} U(t)$. We then obtain:

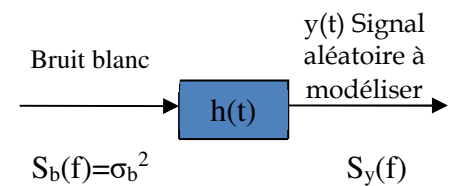
$$-\mu_y(t) = \mu_x H(0) = 0 \times \frac{5}{2+2\pi j f} \Big|_{f=0} = 0 \times 5/2 = 0$$

$$-S_y(f) = |H(f)|^2 S_x(f) = \left| \frac{5}{2+2\pi j f} \right|^2 \cdot TF\left(\sigma^2 e^{-\frac{|\tau|}{\theta}}\right) = \frac{25}{4+(4\pi^2 f^2)} \cdot \frac{2\sigma^2 \theta}{1+4\pi^2 f^2 \theta^2}$$

Example 2: Consider for example the case where the input signal is a white noise $b(t)$. Its DSP is therefore a constant function. Then, at output, we will have a signal such that: $S_y(f) = \text{constant} |H(f)|^2$

Concept of filter trainer

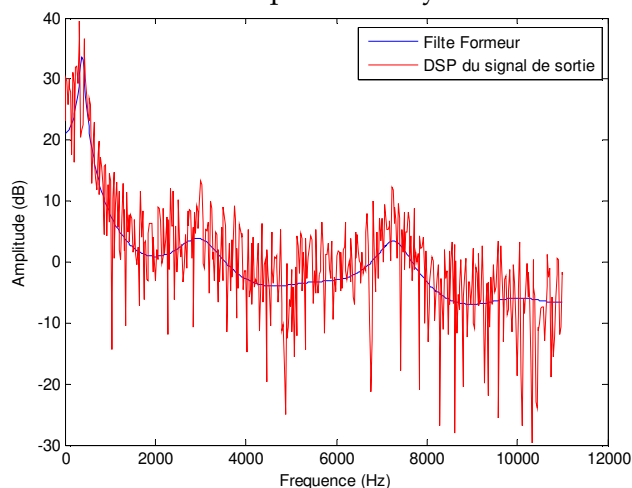
Suppose given a random signal $y(t)$: We call forming filter of $y(t)$; the transfer function filter $H(f)$; such that $y(t)$ is generated by passing a white noise $b(t)$ into $H(f)$.



The shaping filter is determined by reversing the previous formula:

$$|H(f)|^2 = S_y(f) / \sigma_b^2$$

We can therefore describe the random signal by the parameters of the filter and the variance of the white noise. Indeed, if we pass the white noise through a linear and stationary filter with adjustable parameters and if we obtain the desired signal (to be modeled) at the filter output, then we can say that all the spectral information is contained in the filter represented by its coefficients (See exo 1 of LW n°3)



Note : Thus white noise plays for random the equivalent of the Dirac distribution for deterministic.

Remember that white noise has no physical existence because it would be of infinite power. An approximation of band-limited white noise, also called colored white noise, is defined by:

$$S_b(f) = \begin{cases} \sigma_b^2 & |f| < B \\ 0 & \text{ailleurs} \end{cases}$$

- **Application 1: Matching filtering and optimal filtering**

The transmission of a signal is accompanied by distortions (due to the transmission media) which it would be desirable to eliminate or at least attenuate before any subsequent processing. If we know the original (deterministic) signal, we speak of detection. Otherwise, or if the signal is random, the term estimation will be used. There are many detection approaches, one can, among other things, proceed by filtering which will make it possible to enhance the signal drowned in the noise.

Let us consider a deterministic signal $x(t)$ assumed to be known, whose possible presence we wish to test in an observation $s(t)$. The observation noise is assumed to be stationary SSL with spectral density $S_b(f)$. We are looking for a filter $H(f)$ which maximizes the SNR at a precise time T_0 .

We therefore assume that the useful signal $x(t)$ is buried in an additive SSL stationary noise $b(t)$, hence:

$$s(t) = x(t) + b(t)$$

The signal is filtered by a linear filter whose impulse response is $h(t)$. At the filter output, we obtain a signal:

$$y(t) = s(t) * h(t) = x(t) * h(t) + b(t) * h(t) = x_2(t) + b_2(t)$$

At the output of the filter and at time T_0 , the SNR is written : $SNR(T_0) = \frac{Puis(x_2(T_0))}{Puis(b_2(T_0))}$

The goal being to find $h(t)$, we will then express the SNR as a function of $h(t)$ (or $H(f)$), as follows:

- The signal at the numerator $x_2(t)$ is deterministic so its power is expressed as follows:

$$Puis(x_2(T_0)) = |x_2(T_0)|^2 = |TF^{-1}(X_2(f))|^2 = \left| \int X(f)H(f)e^{2\pi j f T_0} df \right|^2$$

- The signal $b_2(t)$ comes from the filtering of a random signal $b(t)$ SSL so it is also SSL, so its power can be formulated as follows:

$$Puis(b_2(T_0)) = E\{b_2(T_0)^2\} = R_{b_2}(\tau=0) = TF^{-1}(S_{b_2}(f))\Big|_{\tau=0} = \int S_b(f)|H(f)|^2 df$$

Thus, we can express the SNR as follows:

$$SNR(T_0) = \frac{P_{uis}(x_2(T_0))}{P_{uis}(b_2(T_0))} = \frac{\left| \int X(f)H(f)e^{2\pi jfT_0} df \right|^2}{\int S_b(f)|H(f)|^2 df}$$

$$SNR(T_0) = \frac{\left| \int a(f)b^*(f)df \right|^2}{\int a(f)a^*(f)df}$$

Which can also be written as:

$$\text{with } a(f) = \sqrt{S_b(f)}H(f)$$

$$b(f) = X^*(f)e^{-2\pi jfT_0} / \sqrt{S_b(f)}$$

To find $H(f)$ which maximizes this ratio, we use the Cauchy-Schwartz inequality:

$$\left| \int a(f)b^*(f)df \right|^2 \leq \int a(f)a^*(f)df \int b(f)b^*(f)df$$

$$\text{Which gives us : } SNR(T_0) = \frac{|\int a(f)b^*(f)df|^2}{\int a(f)a^*(f)df} \leq \int b(f)b^*(f)df$$

with equality (i.e. the maximum) when $a(f)=kb(f) \Rightarrow$ The latter provides us with the following expression of the optimal filter $H(f)$:

$$H(f)_{optimal} = k.X^*(f)e^{-2\pi jfT_0} / S_b(f)$$

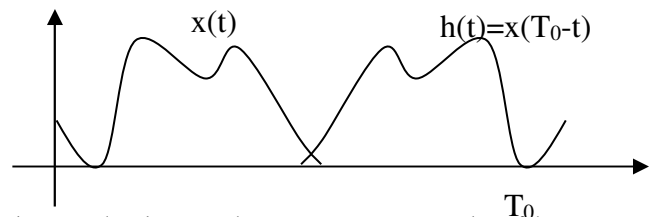
The expression of the SNR becomes: $SNR(T_0) \int \frac{|X(f)|^2}{S_b(f)} df_{max}$ (This maximum is independent of T_0)

Special case : Furthermore, if the noise $b(n)$ is white, we speak of an Matching filter:

$$H(f)_{adapt\acute{e}l} = k/\sigma_b^2 X^*(f)e^{-2\pi jfT_0} \quad \text{giving} \quad h(t)_{adapt\acute{e}l} = k/\sigma_b^2 x^*(T_0 - t)$$

$$SNR(T_0) \frac{E_x}{\sigma_b^2_{max}}$$

The impulse response of the filter represents the useful signal $x(t)$ reversed and shifted by T_0 . Matching filtering amounts to carrying out the cross-correlation between the observation and the signal to be detected.



This response is not causal, which does not allow it to be applied in real time. However, this filter can be applied to a saved signal.

Example 1 : Detection of a pulse

Consider a system emitting a rectangular pulse $x(t)$ of duration T_0 and amplitude A . The additive noise is white. The filter will have the same expression, i.e. $h(t)=x(t)$.

Example 2: Using sonar or radar, we seek to locate a “target”. This target can for example be a boat or a plane. To do this, we proceed as follows: we emit a signal $x(t)$, which travels the distance d to the target, on which it will be reflected towards a receiver. The receiver then receives the noisy signal $y(t)$, attenuated (from a) and delayed by T_{AR} (see LW n°3)

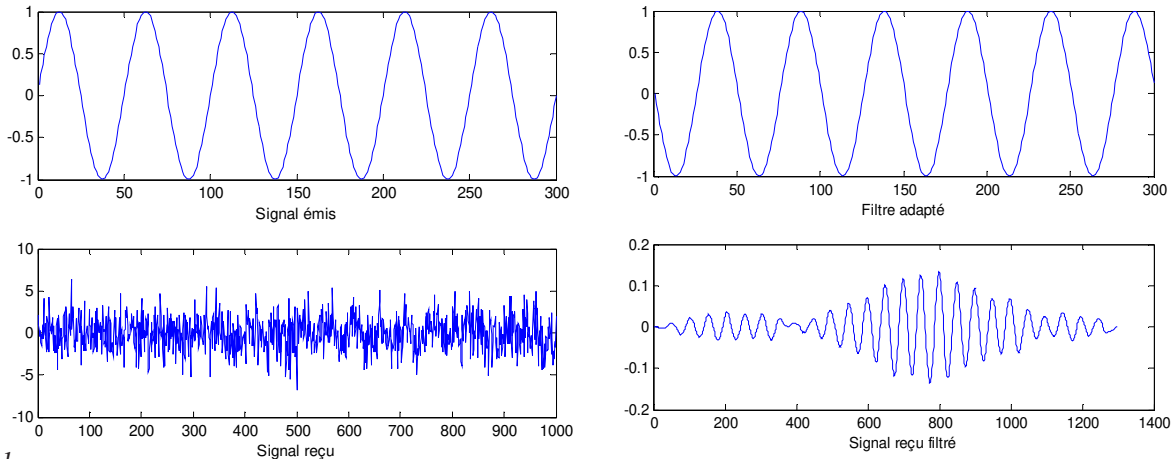
$$y(t) = ax(t - T_{AR}) + b(t). \text{ where } T_{AR} \text{ corresponding to the round trip time (} T_{AR} = 2d/v \text{)}$$

We will then use as matched filter the filter matched to $x(t)$, i.e. $h(t) = x^*(T_0 - t)$

The output of this filter will be the cross-correlation of the signals $y(t)$ and $x(t)$. We then obtain:

$$R_{yx}(tT_0) = aR_{xx}(t - T_0 - T_{AR}) + R_{bx}(tT_0).$$

Knowing that the autocorrelation is maximum at 0, the cross-correlation will be maximum for $t = T_0 + T_{AR}$ which will allow us to determine T_{AR} .



Remarks

- The choice of signal 'x' is important: It is desirable that the max of its autocorrelation function R_{xx} be easy to identify (clearly visible peak as for the triangle function)
- If the signal is sinusoidal, the matched filter is an ideal band pass centered on the frequency of the signal (synchronous detection).

Application Exercise (Exam 2014/2015)

Let $x(t) = 1 - t/T$ with $t \in [0, T]$

This signal is used to determine the distance of an object. The signal received by the receiver after reflection on the object is: $y(t) = \alpha x(t - T_D) + b(t)$ where α is a positive real constant and $b(t)$ is a white noise of power spectral density σ^2 . We want to maximize the signal to noise ratio

1. Determine the impulse response of the filter $h(t)$ such that $\int h(t)^2 dt = 1$ (take $T_0 = T$).
2. Give the signal to noise ratio after filtering.
3. Express the temporal cross-correlation $R_{yx}(tT_0)$ as a function of the auto-correlation of $x(t)$ and the cross-correlation of the signals $b(t)$ and $x(t)$
4. How to determine the time T_D ?
5. Why can't we use optimal filtering for a deterministic signal whose expression we don't know?

• Application 2: Wiener Filtering

It is part of the family of linear estimators with minimum variance which consist in seeking an estimate \hat{y} of y which is a linear function of the observations, that is to say: $\hat{y} = \sum_{i=1}^n h_i x_i = h^T x$

In the case of root mean square estimation, the filter h must then be determined such that the variance of the estimation error $E\{(y - \hat{y})^2\}$ is minimal:

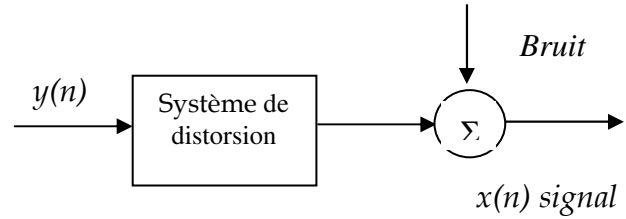
$$E\{(y - \hat{y})^2\} = E\{(y - h^T x)^T (y - h^T x)\} = E\{yy^T\} - E\{y^T h^T x\} - E\{hx^T y\} + E\{hx^T h^T x\}$$

We start by differentiating with respect to h and then we cancel the derivative, which gives us

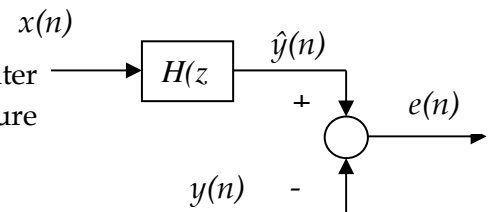
$$-E\{y^T x\} - E\{x^T y\} + 2h^T E\{x^T x\} = 0 \Rightarrow h = E\{y^T x\} / E\{x^T x\} = R_{yx} / R_{xx}$$

In many applications, the time signals are marred by a noise that it is desired to eliminate or at least reduce. As the random useful signal occupies the same frequency bands as the parasitic signal, conventional filtering cannot be used. The Wiener filter provides a solution to this problem when the process is stationary. Wiener filters are said to be optimum in the sense of the criterion of the mean square error between their output and a desired output. .

The following figure illustrates a common linear estimation problem. $y(n)$ corresponds to the stationary random signal which interests us but is not directly accessible. Only $x(n)$ is, it is obtained after passing $y(n)$ into a linear system followed by the addition of a stationary random noise $bb(n)$.



The problem that arises is how to find $y(n)$ from $x(n)$. One solution is to filter $x(n)$ such that the output $\hat{y}(n)$ is as close as possible to $y(n)$. We can measure the quality of the estimate by $e(n)$ defined by: $e(n) = y(n) - \hat{y}(n)$



We therefore seek a filter which minimizes the error. It is convenient to seek to minimize $e^2(n)$ because it is an easily differentiable quadratic function. Moreover, given that the signals are random, the cost function to be minimized is the root mean square error (MSE) defined by: $\xi(n) = E(e^2(n))$

If it is assumed that the filter sought H is an FIR filter of length N (of order $N-1$), the coefficients thereof can be calculated by solving a linear system of equations. $h = [b_0 \ b_1 \ \dots \ b_{N-1}]^T$

The estimated signal $\hat{y}(n)$ can then be written: $\hat{y}(n) = \sum_{i=0}^{N-1} b_i x(n-i)$

Remember that we want to minimize $\xi(n) = E\{e^2(n)\} = E\left\{\left(y(n) - \sum_{i=0}^{N-1} b_i x(n-i)\right)^2\right\}$

To obtain the minimum, it suffices to seek to derive and cancel the cost function with respect to the variables b_i of the impulse response of the filter. The derivative of the cost function with respect to the j th point of the impulse response is given by :

$$\begin{aligned} \frac{\partial \xi}{\partial b_j} &= E\left\{\frac{\partial}{\partial b_j}\{e^2(n)\}\right\} = E\left\{2e(n)\frac{\partial e(n)}{\partial b_j}\right\} = E\left\{2e(n)\frac{\partial}{\partial b_j}\{-b_j x(n-j)\}\right\} \\ \frac{\partial \xi}{\partial b_j} &= -E\{2e(n)x(n-j)\} = -E\left\{2\left(y(n) - \sum_{i=0}^{N-1} b_i x(n-i)\right)x(n-j)\right\} \end{aligned}$$

Assuming that the signals $x(n)$ and $y(n)$ are stationary, and canceling the derivative, we find: $R_{yx}(j) = \sum_{i=0}^{N-1} b_i R_{xx}(j-i)$

Which for the different values of j , gives us the following system of equations to solve:

$$\begin{bmatrix} R_{xx}(0) & R_{xx}(1) & \dots & R_{xx}(N-1) \\ R_{xx}(1) & R_{xx}(0) & \dots & R_{xx}(N-2) \\ \vdots & \vdots & \ddots & \vdots \\ R_{xx}(N-1) & R_{xx}(N-2) & \dots & R_{xx}(0) \end{bmatrix} \cdot \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{N-1} \end{bmatrix} = \begin{bmatrix} R_{yx}(0) \\ R_{yx}(1) \\ \vdots \\ R_{yx}(N-1) \end{bmatrix}$$

It should be noted that the obtaining of the coefficients of the filter is based on the knowledge of the autocorrelation function of the input signal and of the cross-correlation between the desired input and output signals.

Example 1: We assume that the observation $x(n)=y(n)+bb(n)$ and that the additive noise $bb(n)$ is centered and not correlated to the signal. Let's simplify the Wiener-Hopf equations accordingly:

$$R_{yx}(k)=E\{y(n)[y(nk)+bb(nk)]\}=R_{yy}(k)$$

$$R_{xx}(k)=E\{(y(n)+bb(n))(y(nk)+bb(nk))\}=R_{yy}(k)+R_{bb}(k)$$

Consider the following system to be solved:

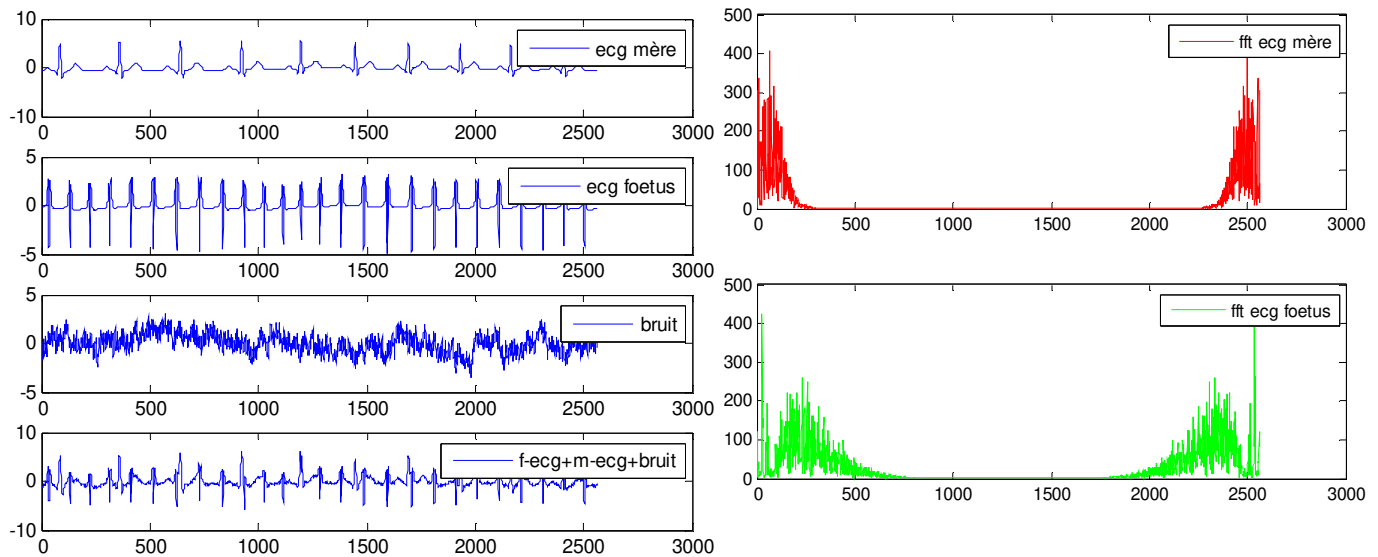
$$\begin{bmatrix} R_{xx}(0) & R_{xx}(1) & \dots & R_{xx}(N-1) \\ R_{xx}(1) & R_{xx}(0) & \dots & R_{xx}(N-2) \\ \vdots & \vdots & \ddots & \vdots \\ R_{xx}(N-1) & R_{xx}(N-2) & \dots & R_{xx}(0) \end{bmatrix} \cdot \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{N-1} \end{bmatrix} = \begin{bmatrix} R_{xx}(0) - R_{bb}(0) \\ R_{xx}(1) - R_{bb}(1) \\ \vdots \\ R_{xx}(N-1) - R_{bb}(N-1) \end{bmatrix}$$

Example 2: We assume that the observation $x(n) = y(n) + bb(n)$

The signal to be estimated $y(n)$ has for the autocorrelation function $R_y(k) = \alpha^{|k|}$, $0 < \alpha < 1$. It is decorrelated from the white noise $bb(n)$ of variance σ_b^2 . Find $h(n)$ such that $H(z) = b_0 + b_1 z^{-1}$

$$\begin{bmatrix} 1 + \sigma_b^2 & \alpha \\ \alpha & 1 + \sigma_b^2 \end{bmatrix} \cdot \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \alpha \end{bmatrix} \quad H(z) = \frac{1}{(1 + \sigma_b^2)^2 - \alpha^2} [(1 + \sigma_b^2 - \alpha^2) + \alpha \sigma_b^2 z^{-1}]$$

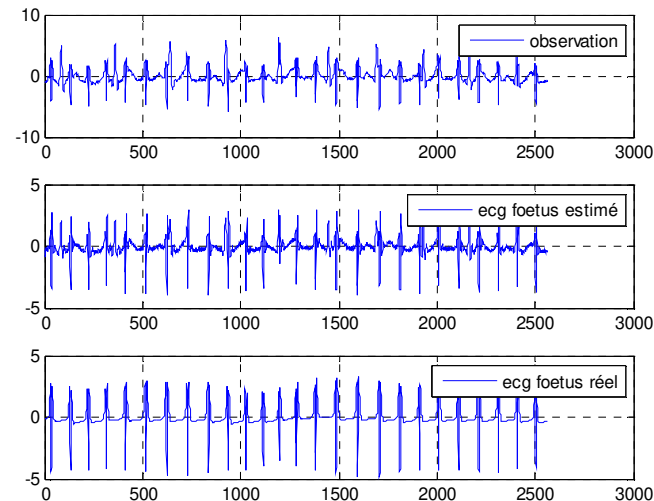
Example 3: As an illustration of Wiener filtering, the cardiac activity of a fetus is measured using an electrocardiogram (ECG) taken from the mother's abdomen (the signal $x(n)$) and which will naturally be disturbed by the ECG of the latter to which is added the thermal noise of the electrodes and electronic equipment. To find the ecg of the fetus, a second measurement is carried out providing the ECG of the mother (the signal bb). We can then use the Wiener filter to estimate the signal $y(n)$ representing the ECG of the fetus.



It can be observed that the TFs of both ecg signals (mother and fetus) occupy the same frequency range. From the auto-correlations of the ECG measured on the abdomen $x(n)$ and that of the mother $bb(n)$, we find the parameters b_i of the filter that we will apply to $x(n)$ to obtain an estimate of $y(n)$ being the fetal ECG.

Noticed

When the auto and cross-correlation functions are not known (the most common case), then we will approach the optimal Wiener filter using a feedback loop and a minimization algorithm: this is what we call **adaptive filtering**. In this case, we will replace the knowledge of the correlation functions with a learning phase making it possible to iteratively modify the impulse response of the filter.



Exercices n°2 : Random processes

1. Let $X(t)$ and $Y(t)$ be two independent broad-sense stationary (WSS) random signals such that $R_X(\tau) = 9e^{-a|\tau|}$ and $R_Y(\tau) = b\delta(\tau) + c$. We consider $Z(t) = X(t) + Y(t)$,

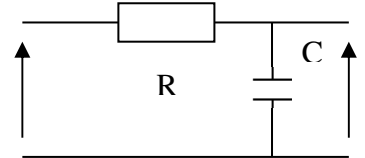
- What conditions must be imposed on the constants a , b and on c (Justify)
- Compute the first-order statistics of $Z(t)$
- Calculate $R_Z(t, \tau)$. Is $Z(t)$ WSS?
- Can we calculate its PSD $S_Z(f)$? if so calculate it.

2. Let the random process be $Z(t) = X\cos(2\pi f_0 t) - Y\sin(2\pi f_0 t)$, where X and Y are random variables centred with a variance of σ^2 ; f_0 is a constant.

- Calculate the mean value $E\{Z(t)\}$ and the variance σ_Z^2 .
- Calculate the autocorrelation function $R_Z(t_1, t_2)$ and examine whether the process is WSS.

3. The input $x(t)$ of the circuit is a white noise of autocorrelation function $R_x(\tau) = \sigma_x^2 \delta(\tau)$

- Determine the spectral density of the output $y(t)$, denoted $S_y(f)$.
- Determine the autocorrelation function of the output $y(t)$, denoted $R_y(\tau)$ as well as its power.
- Replace R with a self L and C with a resistor and repeat the questions.



4. The signals $x(n)$ and $y(n)$ were obtained by filtering, by means of a finite impulse response filter, a centered Gaussian white noise $b(n)$ of variance σ^2 .

A] The filtering equation is: $x(n) = 2.b(n) + 0.5.b(n-1) - 0.2.b(n-2) + 0.1.b(n-3)$

- Calculate, as a function of σ^2 , the autocorrelation coefficients of order 0,1,2,3 of the signal $x(n)$.
- We will note $R_{xx}(0), R_{xx}(1), R_{xx}(2), R_{xx}(3)$ for these coefficients.
- Is the distribution of the amplitude levels of the signal $x(n)$ Gaussian (without justifying)?

B] $y(n) = b(n) - b(n-2)$

- Give the ZT of the impulse response of the filter which made it possible to obtain $y(n)$ from $b(n)$.
- Place the zeros of this filter on a unit circle, what are the frequency(ies) cut by this filter?
- Plot approximately its frequency response.

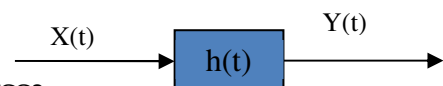
C] We now consider the cross-correlation coefficient $R_{xy}(k)$, between the signals $x(n)$ and $y(n)$ given by $R_{xy}(k) = E[x(n)y(n-k)^*]$.

Calculate $R_{xy}(0), R_{xy}(1), R_{xy}(2), R_{xy}(-1), R_{xy}(-2)$

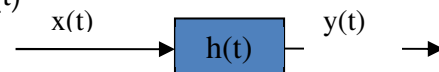
5. Let $X(t) = A + b(t)$ be a real random signal, where A is a real constant and $b(t)$ is white noise with power spectral density σ_b^2 , and let there be an averaging filter with impulse response:

$$h(t) = \frac{1}{T} \Pi(t - T/2)$$

- Express the statistical autocorrelation of the input signal $R_{xx}(t, \tau)$. Is $X(t)$ WSS?
- Determine the statistical mean of $y(t)$.
- Show that $R_{yy}(\tau) = \frac{\sigma^2}{T} \Lambda_T(\tau) + A^2$ and deduce σ_y and μ_y .
- Take $A=2$ and plot $X(t)$ and $Y(t)$.



6. We consider the diagram opposite where $x(t) = s(t) + b(t)$ with $s(t) = \frac{\Pi(t)}{1}$ and $b(t)$ is a centered white Gaussian noise with variance 1.



- Determine the probability density of $x(t)$. Is $x(t)$ WSS?
- We want to maximize the signal-to-noise ratio (SNR)
- Determine and plot $h(t)$ for $k=2$ and $T_0=2$. Then calculate the SNR.
- Which probability distribution follows $y(t)$ (justify).
- Give a concrete example of the use of this type of filter
- If $s(t)$ is deterministic and unknown, what filter do we use?

7. We consider the transmission of two symbols:

$$s_0(t)=A \quad t \in [0, T], \quad s_1(t)=-A \quad t \in [0, T]$$

through a Gaussian additive white noise channel. The signal received is written: $x(t)=s_i(t)+b(t)$ where $b(t)$ is a centered white noise of DSP σ_b^2

- Determine the impulse response of the matched filter $h(t)$ corresponding to $s_0(t)$ such that $\int h(t)^2 dt=1$
- Same question for $s_1(t)$. Then, give the signal-to-noise ratio, in each case.

We note $y_{si}(t)$, the output of the filter corresponding to $s_i(t)$,

$$\text{- Show that for } b(t) \text{ it is written: } y_b(T) = \frac{1}{\sqrt{T}} \int_0^T b(t) dt$$

- Determine the mean and variance of $y_b(T)$
- Assuming that the white noise is Gaussian, determine the probability distribution of the random variable

$$Y = y_{si}(t) + y_b(t)$$

8. Consider a process of the form $X(n) = \theta + W(n)$ where $W(n)$ is a Gaussian process with mean 0 and variance 1 such that $W(n)$ and $W(j)$ are independent if $n \neq j$. We assume that θ follows a law $N(0, \sigma^2)$ independent of $W(n)$ $n \in \mathbb{Z}$.

- Characterize the Wiener filter allowing θ to be estimated from $X_n, X_{n-1}, \dots, X_{n-N+1}$.
- Calculate the filter coefficients.

9. It is desired to de-noise a speech signal $z(n)$ corrupted by additive noise $b(n)$ independent of the sound signal and this, by Wiener filtering. We assume that some autocorrelation values are known for the two signals such that:

$$R_{ZZ}[0] = 1.5; R_{ZZ}[1] = 0.5; R_{ZZ}[2] = 0.25; R_{ZZ}[3] = 0.125; R_{ZZ}[4] = 0.0625$$

$$R_{bb}[0] = 1; R_{bb}[1] = 0.25; R_{bb}[2] = 0.0625; R_{bb}[3] = 0.015625$$

- Why can't we use a matched filter or an averaging filter?
- Give the Wiener-Hopf equations to estimate $z(n)$.
- Determine the Wiener filter of order 2 allowing to find the useful signal $\hat{z}(n)$ then express $H(z)$.

10. Consider a noise estimation problem $b(n)$.

The observed signal is $x(n) = s(n) + b(n) - b(n-1)$.

It is assumed that the statistical correlation of the signal $s(n)$ is $R_{ss}(n) = 0.8^{|n|}$ and that it is decorrelated from the noise, the autocorrelation of which is $R_{bb}(n) = 0.8 \delta(n)$.

- Determine the statistical means of $s(n)$ and $b(n)$.
- When is the Wiener filter used?
- Give the Wiener-Hopf equations for estimating $b(n)$
- Determine the Wiener filter of order 2 making it possible to find the useful signal $\hat{b}(n)$.
- Express $\hat{b}(n)$

Solutions

1. See test 1 2016/2017

$$2. \mu_z(t) = 0 \quad \sigma_z^2(t) = \sigma^2 R_z(\tau) = \sigma^2 \cos(2\pi f_0 \tau) \text{ WSS}$$

$$3. R_{xx}(t, \tau) = A^2 + \sigma_b^2 \delta(\tau) = \text{fct}(\tau) \quad \mu_x(t) = A \Rightarrow \text{WSSL} \quad H(f) = \text{sinc}(fT) e^{-\pi j f T} \quad \mu_y(t) = A \text{ averaging filter}$$

4. $H(f) = 1/(1 + j2\pi fRC)$, $S_y(f) = \sigma_x^2 / (1 + 4(\pi fRC)^2)$, $R_y(\tau) = \sigma_x^2 / 2RC e^{-|\tau|/RC}$, $P = R_y(0) = \sigma_x^2 / 2RC$, interro1 16/17
5. $R_x(0) = (b_0^2 + b_1^2 + b_2^2 + b_3^2)$, $\sigma^2 R_x(1) = (b_0 b_1 + b_1 b_2 + b_2 b_3)$, $\sigma^2 R_x(2) = (b_0 b_2 + b_1 b_3)$, $\sigma^2 R_x(3) = (b_0 b_3)$, $\sigma^2 R_x(k \geq 4) = 0$
 $H(z) = (z^2 - 1)/z^2$, $R_{xy}(0) = 2.2$, $R_{xy}(1) = 0.4$, $R_{xy}(2) = -0.2$, $R_{xy}(3) = 0.1$, $R_{xy}(-1) = -0.5$, $R_{xy}(-2) = -2$
6. $x(t)$ Gaussian with mean $s(t)$ and variance 1. $x(t)$ non stationary $S_b(f) = |H(f)|^2 S_s(f) = |H(f)|^2 \text{sinc}^2(f)$, $h(t) = 2\pi(2 - t)$, $\text{SNR} = 1$, $y(t)$ Gaussian Radar or Sonar Low pass (average)
7. Question1 14/15 8. $R_{xx}(0) = 1 + \sigma^2$, $R_{xx}(k > 0) = \sigma^2 R_{\theta x}(k) = \sigma^2 b_i = \sigma^2 / (N\sigma^2 + 1)$ 9. $\mu_z = 0$, $\mu_b = 0$, random (speech) signal,
10. $\mu_s = 0$, $\mu_b = 0$, $S_b(f) = 0.8$. When useful signal and noise occupy the same frequency range. $b_0 = 0.8/2.6$ and $b_1 = 0$.

Additional exercises

1. Consider the random signal defined by: $x(t) = A_1 e^{j2\pi f_1 t} + A_2 e^{j2\pi f_2 t}$ where A_1 and A_2 are two Gaussian variables, uncorrelated, centered and of variance σ^2
- Give the joint ddp $f_{A_1, A_2}(A_1, A_2)$
 - Determine the statistical mean of the signal $x(t)$ as well as its autocorrelation function and the DSP
 - Determine the ddp of $x(t)$
 - Is the process $x(t)$ stationary at the 2nd order?

2. The real WSS random signal $x(t)$ has an autocorrelation function of the form $R_x(\tau) = \sigma_x^2 e^{-\beta|\tau|}$. Another signal is related to $x(t)$ by the following deterministic equation: $y(t) = ax(t) + b$, where a and b are given constants.

- What is the autocorrelation function of $y(t)$? Deduce μ_y and σ_y^2
- What is the cross-correlation and covariance function of $x(t)$ and $y(t)$?
- Calculate the correlation coefficient. Was this result predictable, why?

3. Which of the following satitistic autocorrelation functions can be those of a real random process?

$R_{x1}(\tau) = \Lambda_2(\tau) - 2$	$R_{x2}(\tau) = -\Lambda_2(\tau) + 2$	$R_{x3}(\tau) = e^{-2 \tau }$	$R_{x4}(\tau) = e^{-2\tau} U(\tau)$	$R_{x5}(\tau) = \delta(\tau - 1) + \delta(\tau + 1)$
--------------------------------------	---------------------------------------	-------------------------------	-------------------------------------	--

4. We consider the signal $x(n) = g(n)\cos(2\pi k_0 n/N + \phi)$ defined for $n \in [0, N-1]$, and where $g(n)$ is a random function SSL, independent of ϕ uniform va ente 0 and 2π .

- Calculate the autocorrelation of $x(n)$
- Deduce into its power spectral density, as a function of the PSD of g , $S_g(f)$.

5. Consider the random signal $x(t)$ WSS whose DSP is given by $S_x(f) = \sigma^2 B \Pi_B(f)$

- Determine the statistical autocorrelation of $x(t)$
- Deduce the mean and the statistical variance of $x(t)$.

This signal is transmitted through a LTI whose transfer function $H(f) = \Pi_A(f)$ with $A < B$

- Is the output signal random? (Justify) SSL? (Justify)
- Determine $S_y(f)$ and deduce $R_y(\tau)$ then the statistical moments of order 1 of $y(t)$.

6. We have a received signal which is the noisy, delayed and attenuated version of a signal of interest $s(n)$. The noise $b(n)$ is assumed Gaussian white of variance σ^2 . The problem is to determine the amplitude A and the delay n_0 in the received signal $x(n) = As(n - n_0) + b(n)$. Knowing that the signal-to-noise ratio is maximum at the output of the response matched filter $h(n) = s(-n)$.

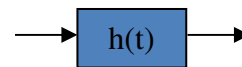
- Verify that the output $y(n)$ of this filter is expressed as the sum of two correlation functions.

- Calculate the variance σ_b^2 of the output noise.
- We take for $s(n)$ a rectangular pulse of width L , then plot an example of received signal and matched filter output.

7. Let $x(t) = t$ with $0 \leq t \leq T$

This signal is used to determine the distance of an object. Knowing that the signal received $y(t)$ by the receiver is delayed by T' and noisy by white Gaussian noise of power spectral density σ^2 :

1. Plot approximately $y(t)$.
2. Determine the general expression for the impulse response of the filter $h(t)$ (whose energy is 1) allowing the signal-to-noise ratio to be maximized.
3. Plot $h(t)$ as a function of $T_0 = 2T$ and $k = \sigma^2$, then $z(t)$ the filter output (take $T' = 10$).
4. Give the signal to noise ratio after filtering.
5. Why is this filtering said, in general, optimal and, in particular, adapted?



8. We consider a process of the form $X(n) = b(n) + \alpha b(n-1) + W(n)$ where $W(n)$ is a Gaussian process with mean 0 and variance σ^2 such that $W(n)$ and $W(j)$ are independent if $n \neq j$. We assume that $b(n)$ is a uniform random variable with values in $\{-1, 1\}$, independent of $W(n)$ $n \in \mathbb{Z}$ and likewise, $b(n)$ and $b(j)$ are independent if $n \neq j$ ($P(b(n) = 1) = P(b(n) = -1) = 1/2$).

- Build a 3rd order filter that estimates $b(n)$.

9. We consider a problem of estimation of a noisy signal $s(n)$.

The observed signal is $x(n) = \alpha s(n) + b(n)$.

We assume that $s(n)$ and $b(n)$ are WSS and decorrelated and that the noise has a PSD $S_{bb}(f) = 0.25$.

- Remind the application conditions of Wiener filtering
- Determine the Wiener filter of order 2 making it possible to find the useful signal $\hat{s}(n)$. We will assume that $R_{ss}(k) = 2 \cdot 0.5^{|k|}$
- Express $H(z)$ then $\hat{s}(n)$

10. We consider a problem of estimation of a signal $s(n)$ noisy and having undergone an echo.

The observed signal is $x(n) = s(n) + 0.5s(n-1) + b(n)$.

We assume that the autocorrelation of the useful signal is known and that it has the expression $0.5^{|k|}$ and it is assumed that the useful signal is decorrelated from the noise whose autocorrelation is $R_{bb}(k) = 0.25^{|k|}$.

1. Determine the statistical means of $x(n)$ and $b(n)$
2. Give the Wiener-Hopf equations for estimating $s(n)$
3. Determine the Wiener filter of order 2 making it possible to find the useful signal $\hat{s}(n)$.
4. Express $\hat{s}(n)$ and comment

Solutions

1. See test 1 2014/2015 2. $R_y(\tau) = \sigma_x^2 e^{-\beta|\tau|} + b^2 S_y(f) = 2\beta\sigma_x^2 / (\beta^2 + 4(\pi f)^2) + b^2 \delta(f)$ $R_{xy}(\tau) = a R_x(\tau)$

3. See question 1 2015/2016 4. $R_x(m) = R_g(m) \cos(2\pi k_0 m/N) / 2$ $S_x(f) = [S_g(f-k_0) + S_g(f+k_0)] / 4$

5. $R_x(\tau) = \sigma^2 B^2 \text{sinc}(B\tau)$, $\mu_x = 0$, $\sigma_x^2 = \sigma^2 B^2$, $x(t)$ WSS $\Rightarrow y(t)$ WSS, $S_y(f) = \sigma^2 B^2 \Pi_A(f)$, $R_x(\tau) = \sigma^2 AB \text{sinc}(A\tau)$

6. $y(n) = AR_s(n-n_0) + R_{bs}(n)$ $\sigma_b^2 = R_{b'}(0)$ $S_{b'}(f) = \sigma_b^2 |H(f)|^2 \Rightarrow R_{b'}(0) = \sigma_b^2 \int_{-1/2}^{1/2} |H(f)|^2 df$

7. Review 15/16 8. $R_{xx}(1:3) = (1 + \alpha^2 + \sigma^2, \alpha, 0)$ $R_{bx}(1:3) = (1, 0, 0)$

9. Signals: known useful and stationary and conjointly stationary observation

$R_{xx}(k) = \alpha^2 R_{ss}(k) + R_{bb}(k)$ $R_{sx}(k) = \alpha R_{ss}(k)$

Laboratory Work n° 3 : Random processes and spectral estimation

Goals: Manipulate random signals, characterize them using their moments of order 1 (mean, variance) and order 2 (autocorrelation, covariance). Approach and acquire the notions of stationarity and ergodicity. Spectral estimation. Address the notion of Forming Filter.

1. Demo

```
# -*- coding: utf-8 -*-
import numpy as np; import matplotlib.pyplot as plt;
import scipy.signal as sp
N_R=500; N_va=1000; A=1; f0=.002; X=[];

t = np.linspace(0,N_va-1,N_va)
phi=np.random.uniform(0,2*np.pi,N_R);

for i in range (N_R):
X=np.append(X,A*np.cos(2*np.pi*f0*t+phi[i]))
X= np.resize(X,new_shape=(N_R,N_va))

plt.figure(1);plt.plot(t,X[0,:]);plt.plot(t,X[1,:]);plt.plot(t,X[2,:]);plt .plot(t,X[3,:]);
"""Stationarity of order 1"""
avg=np.mean(X,0); var=np.var(X,0);
plt. figure(2); plt.subplot(211); plt.plot(avg); plt.subplot(212); plt. plot(var);
"""Stationarity of order 2"""
Cx=np.cov(X, rowvar=False);
plt. figure(3); plt. imshow(Cx);
plt. figure(4); plt. plot(Cx[0,:]); plt. plot(Cx[20,:]); plt. plot(Cx[40,:]);
"""Ergodism of order 1"""
X1=X[1,:];
avg_t=np.mean(X1); var_t=np.var(X1);
"""Ergodism of order 2"""
Cx_t=np.correlate(X1-avg_t,X1-avg_t,'full')/N_va
Rx_t=np.correlate(X1,X1,'full')/N_va
plt.figure(5)
plt.plot(Cx_t,label='Cov temp'); plt.plot(Rx_t,label='Cor temp');
plt.legend()

# A = np.random.uniform(-1,1,N_R);
# for i in range (N_R):
# X=np.append(X,A[i]*np.cos(2*np.pi*f0*t))

#X=np.random.randn(N_R,N_va)

# mux=0; varx=1;
# X=mux+np.sqrt(varx)*np.random.randn(N_R,N_va)

""" 1 realization of Gaussian white noise """
""" Ergodic + stationary hypothesis """
mux=0; varx=1; N=1000;
bb=mux+np.sqrt(varx)*np.random.randn(N)
fe=22050; Te=1/fe;
t = np.linspace(0, N-1, N)*Te;
```

```

Moy_s=Moy_t=np.mean(bb);
Var_s=Var_t=np.var(bb);
Rb_s=Rb_t = np.correlate(bb,bb,mode='same')/N;
tt = np.linspace(1-N/2, N/2-1, N)*Te;
plt.figure(6);plt.subplot(311); plt.plot(t,bb); plt.grid(True);
plt.subplot(312); plt.plot(tt,Rb_s); plt.grid(True);
plt.xlabel('time'); plt.ylabel('Amp'); plt.title('White noise statistical autocor');
NF=2048;
TFb = np.fft.fft(bb,NF); TFb=np.fft.fftshift(TFb);
Sbf = np.abs(TFb)**2/N; ff = np.linspace(-NF/2, NF/2-1, NF)*fe/NF;
s=np.mean(Sbf)
plt.subplot(313); plt.plot(ff,Sbf); plt.grid(True);plt.title("White Noise DSP");

""" Calculation of DSP: Spectral estimation """
f0=5000; X = bb + 0.5*np.cos(2*np.pi*f0*t)
#f1=5050; X = X + 0.5*np.cos(2*np.pi*f1*t)
NF=2048;
TFx1 = np.fft.fft(X,NF)/N; TFx1=np.fft.fftshift(TFx1);
Sxf1 = np.abs(TFx1)**2; ff = np.linspace(-NF/2, NF/2-1, NF)*fe/NF;
Rx_s=Rx_t = np.correlate(X,X,mode='same')/N; Sxf2 = np.fft.fft(Rx_s,NF)/N; Sxf2=np.fft.fftshift(Sxf2); Sxf2 = np.abs(Sxf2)
ff3,Sxf3 = sp.periodogram(X,fs=fe>window='boxcar',nfft=NF,return_onesided=False,scaling='spectrum')
ff4,Sxf4 = sp.periodogram(X,fs=fe>window='hann',nfft=NF,return_onesided=False,scaling='spectrum')
ff5, Sxf5 = sp.welch(X,fs=fe>window='hann',nperseg=N//4, nfft=NF,return_onesided=False,scaling='spectrum')
var_ss=np.mean(Sxf1)
ff = np.linspace(-NF/2, NF/2-1, NF)*fe/NF;
plt.figure(7)
plt.subplot(221);plt.plot(ff,Sxf1,ff3,Sxf3); plt.title('DSP=Rectangular Window Periodogram');
plt.subplot(222);plt.plot(ff,Sxf2); plt.title('Correlogram');
plt.subplot(223);plt.plot(ff4,Sxf4); plt.title('Hanning Window Periodogram');
plt.subplot(224);plt.plot(ff5,Sxf5); plt.title('Average Hanning Window Periodogram');

""" Concept of Shaper filter """
a = np.array([1,-0.839,-0.015,-0.320,0.197,0.055,-0.285,0.067,0.044,0.003,0.178])
b = np.array([1,0]);
L=500; f,H= sp.freqz(b,a,L,fs=fe);
Y = sp.lfilter(b,a,bb);
TFb = TFb[0:L];
plt.figure(8); plt.subplot(211);plt.plot(f, 20*np.log10(abs(TFb/TFb.max())));
plt.title('Input Signal: White Noise'); plt.xlabel('Freq ( Hz)'); plt.ylabel('Amp');
plt.subplot(212); plt.plot(f, 20*np.log10(abs(H/H.max())));
plt.title('Filter Module (blue)+ Filter Output (orange)'); plt.xlabel('Freq ( Hz)'); plt.ylabel('Amp');
TFy = np.fft.fft(Y); TFy = TFy[0:L];
plt.plot(f, 20*np.log10(abs(TFy/TFy.max())));

```

2. To do

Download the 'you have mail waiting.wav' file and place it in the same directory as your program starting as follows:

```

import numpy as np; import matplotlib.pyplot as plt;
from scipy.io import wavfile as wf; import winsound;
fname = 'youhavependingmail.wav';
winsound.PlaySound(fname, winsound.SND_FILENAME)
fe, X = wf.read(fname);
Te=1/fe; N=len(X); t = np.linspace(0, N-1, N)*Te;

```

```
plt.figure(1);plt.subplot(211); plt.plot(t,X); plt.grid(True);  
plt.xlabel('time'); plt.ylabel('Amp'); plt.title('Phrase');  
N1=21000;N2=21500; Y=X[N1:N2] ; ty=np.linspace(N1, N2-1, N2-N1)*Te;  
plt.subplot(212); plt.plot(ty,Y); plt.grid(True);  
plt.xlabel('time'); plt.ylabel('Amp'); plt.title('piece of sentence');  
#zz=np.int8(Y); wf.write("z.wav", fe, zz);  
#winsound.PlaySound("z.wav",winsound.SND_FILENAME);
```

1. Is the signal studied a random process or a realization of a random signal?
2. Calculate the PSD. Is it of statistical or temporal origin?
3. Calculate and visualize the temporal mean and variance for different parts of the signal that overlap for durations of

$N=1000, 500, 250, 125$. For what value of N can we consider the process stationary?

1. Compare the 4 techniques by varying A_1 , A_2 and A_3 and comment.
2. Match the frequencies and observe.
3. Comment on the four techniques for estimating PSD by varying N .
4. What is the advantage and disadvantage of averaging?

Laboratory Work n° 4 : Matching filtering and Wiener filtering

Goals: Optimal filtering

- Detection of a known deterministic signal drowned in white noise → Matching filtering
- Filtering a random SSL signal drowned in SSL noise → Wiener filtering

1. Demo

```
# -*- coding: utf-8 -*-
import numpy as np; import matplotlib.pyplot as plt
"Matching filtering"
N=500; var = 0.16; mu = 0;
x=np.ones(10); Delay = 100
y = np.sqrt(var) * np.random.randn(N) + mu
y[Delay:Delay+len(x)]= y[Delay:Delay+len(x)] + x
#Delay=300; y[Delay:Delay+len(x)]= y[Delay:Delay+len(x)] + x
Ryx = np.correlate(y,x,"full"); Ryx=Ryx[len(x)-1:]
plt.figure(1)
plt.subplot(311); plt.plot(x); plt.title('sent signal')
plt.subplot(312); plt.plot(y); plt.title('signal received')
plt.subplot(313); plt.plot(Ryx); plt.title('Cross-correlation between transmitted signal and received signal')
import scipy.linalg as la; import scipy.signal as sp;
fecg=[]; meg=[];

""" Playing Files """
with open('fecg.txt') as f:
    for i in f:
        fecg.append(float(i))
fecg = np.array(fecg)
with open('mecg.txt') as f:
    for i in f:
        mecg.append(float(i))
mecg = np.array(mecg)
""" Added noise """
N=len(mecg)
var = 0.01; bb=(var**0.5)*np.random.randn(N); obs=fecg+mecg+bb;
"""Determination of the coefficients of the Wiener Filter"""
P=10;
obs=obs-np.mean(obs); mecg=mecg-np.mean(mecg);
Rxx = np.correlate(obs,obs,mode='full')[len(obs)-1:];
Rmecg = np.correlate(mecg,mecg,mode='full')[len(mecg)-1:];
Ryx = Rxx-Rmecg;
C = la.toeplitz(Rxx[0:P]);
B = Ryx[0:P];
b = la.inv(C).dot(B)
"""Filtering"""
a = np.array([1,0]);
y = sp.lfilter(b,a,obs);
plt.figure(2);
plt.subplot(311); plt.plot(fecg); plt.grid(True); plt.title('ecg baby');
plt.subplot(312); plt.plot(obs); plt.grid(True); plt.title('Observation');
```



```
plt.subplot(313); plt.plot(fecg); plt.plot(y); plt.grid(True); plt.title('estimated baby ecg');
```

2. To do

"To do appropriate filtering"

1. Create a signal composed of a random sequence of A and -A symbols with a duration of 10 each separated by 20s

```
import numpy as np
import matplotlib.pyplot as plt
W = np.random.randint(0,2,10); A=5;
W = 2*W-1
W = A*W
x = []; break = np.zeros(20)
for i in W:
```

```
symb = i*np.ones(10)
x = np.append(x,symb)
x = np.append(x, pause)
```

2. Add white noise
3. Use suitable filter
4. Visualize the 3 signals

"To Do Wiener Filtering"

```
import numpy as np; import scipy. signal as sp; import matplotlib.pyplot as plt
import sounddevice as snd
import scipy.io.wavfile as wav
import scipy.linalg as la; import scipy. signal as sp;
fe,x = wav.read('youhavependingmail.wav')
```

1. Create a signal composed of the audio signal + white noise
2. Listen to check for the presence of the echo
3. Determine the coefficients of the Wiener filter
4. Filter
5. Listen filtered signal
6. View the 3 signals: original audio signal, with echo and the filtered one
7. View the 3 signal periodograms: original, noisy audio signal and the one filtered on a small portion

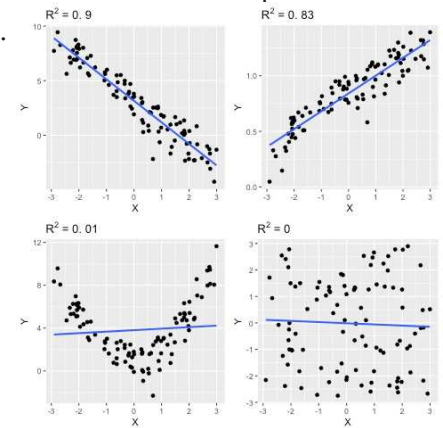
III. Parametric spectral analysis and adaptive digital filtering

Introduction

Parametric modeling consists of associating a model with a signal, represented by a so-called parameter vector $\theta = [\theta_1, \theta_2, \dots, \theta_p]$ supposed to best represent the signal under consideration. It requires the choice of a model which can only be done if one has information on the signal.

It allows, among other things: to represent the data by a vector of smaller dimension. This can be used during compression and transmission, as a signature for identification or classification.

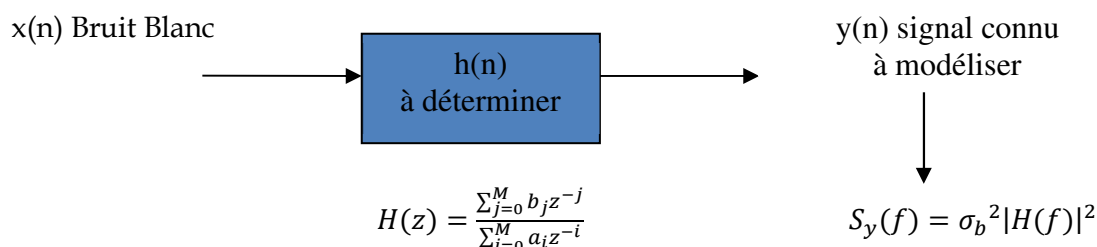
We will approach in this course the case of the spectral modeling which will allow us to estimate the spectrum of the signal.



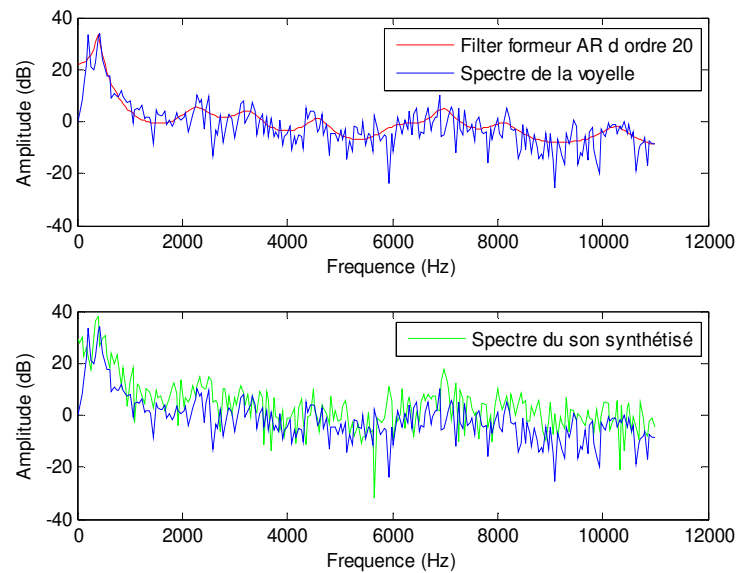
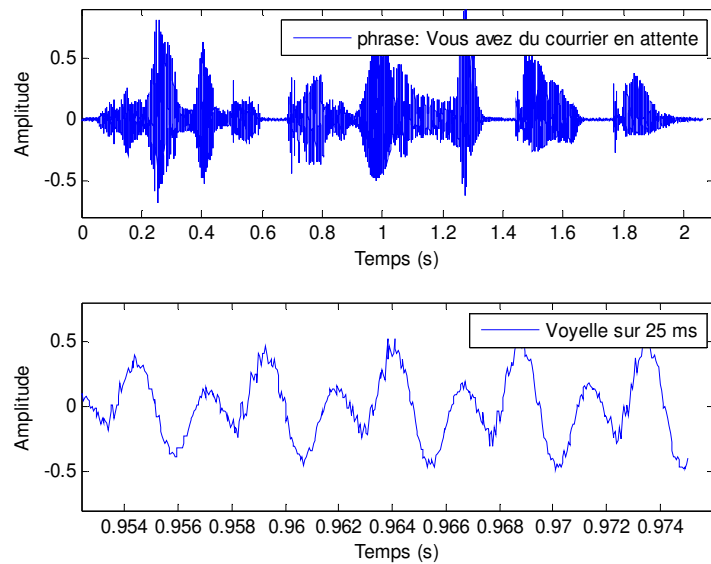
1. Parametric spectral modeling

We saw earlier that a random signal can be modeled (synthesized) as the response of a linear filter to an excitation in the form of white noise such as $|H(f)|^2 = S_x(f)/\sigma_b^2$. This forming filter $H(f)$ is also called a generator process. These parameters associated with the noise variance σ_b^2 constitute the mathematical model corresponding to the random signal. The concept of signal generating process has been particularly developed and applied with digital filters. These are the signal models most used in statistical signal processing (estimate, prediction, etc.). Depending on the nature of the filter, different signal models can be obtained (AR, MA, ARMA, etc.)

Example: During a call by GSM (mobile phone), the mobile which acts as a microcomputer bringing together different functionalities including analysis, synthesis, coding, etc. will allow us to model (random) speech by operating an LPC coding in slices of tens of ms. It consists in finding the parameters of the forming filter $h(n)$ for each slice $y(n)$ recorded and analyzed. It is these parameters (a_i and b_j) which will be transmitted to produce a synthesis signal approaching the original signal $y(n)$.



Below the sentence "you have mail waiting" sampled at a frequency $f_e = 22050$ (45531 samples) and a zoom on a vowel of duration 25 ms (500 samples).



By comparing the appearance of the shaping filter $H(f)$ to that of $S_y(f)$, we note that we find the general appearance of the vowel spectrum, in particular the frequencies whose power is maximum. Knowing that the vowel spectrum has 500 values and that the equivalent filter $H(f)$ is obtained from 20 coefficients, it is better to transmit 21 coefficients ($20 a_i + \text{noise variance}$) than 500 values.

This is why autoregressive models are increasingly used in signal processing: coding and transmission by linear prediction, speech synthesis, recognition, etc.

Note: The speech signal is a long-term non-stationary random process, but it is considered stationary in analysis time windows of the order of 20 to 30ms. This property of short-term stationarity therefore allows progressive analysis and modeling of the speech signal. To avoid any loss of information, care will be taken to take overlapping windows.

2. Auto-regressive (AR) model

Autoregressive signals are obtained by passing white noise through a purely recursive filter. This filter therefore has an infinite impulse response.

$$H(z) = 1 / \left(1 + \sum_{i=1}^N a_i z^{-i} \right)$$

From $H(z)$, we can determine the difference equation:

$$y(n) = x(n) - \sum_{i=1}^N a_i y(n-i)$$

This means that the signal $y(n)$ is assumed to be predictable based on a number of its past values.

Knowing that $x(n)$ is white noise then hence $R_{xx}(k) = \sigma^2 \delta(k)$

$$-\mu_x = E\{x(n)\} = 0 \quad -R_{xx}(0) = E\{x(n)^2\} = \sigma^2 \quad -R_{xx}(k) = E\{x(n)x(n-k)\} = 0 \text{ pour } k \neq 0$$

Let's then calculate $R_{yy}(k)$

$$\begin{aligned}
R_{yy}(k) &= E\{y(n)y(n-k)\} = E\{x(n)y(n-k)\} - \sum_{i=1}^N a_i E\{y(n-i)y(n-k)\} \\
&= E\left\{x(n) \sum_{k'} x(n-k-k') h(k')\right\} - \sum_{i=1}^N a_i R_{yy}(k-i) = \sum_{k'} E\{x(n)x(n-k-k')\} h(k') - \sum_{i=1}^N a_i R_{yy}(k-i) \\
&= \sum_{k'} R_{xx}(k+k') h(k') - \sum_{i=1}^N a_i R_{yy}(k-i) = R_{xx}(k)h(0) + R_{xx}(k+1)h(1) + \dots - \sum_{i=1}^N a_i R_{yy}(k-i)
\end{aligned}$$

-If $k=0$, $R_{yy}(0) = R_{xx}(0) \cdot h(0) - \sum_{i=1}^N a_i R_{yy}(-i) = \sigma^2 \cdot 1 - \sum_{i=1}^N a_i R_{yy}(i)$

- For $k=1$ to N , $R_{yy}(k) = 0 - \sum_{i=1}^N a_i R_{yy}(k-i) = -\sum_{i=1}^N a_i R_{yy}(k-i)$

We can use a matrix form: $R \cdot \underline{a} = \underline{S}$ (Recall that for a real signal $R_{yy}(k) = R_{yy}(-k)$)

$$\begin{bmatrix} R_{yy}(0) & R_{yy}(1) & \dots & R_{yy}(N) \\ R_{yy}(1) & R_{yy}(0) & \dots & R_{yy}(N-1) \\ \vdots & \vdots & \ddots & \vdots \\ R_{yy}(N) & R_{yy}(N-1) & \dots & R_{yy}(0) \end{bmatrix} \cdot \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} \sigma^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

R is the autocorrelation matrix whose general term r_{ij} depends only on the difference ij (Toeplitz matrix). The resolution of these so-called Yule-Walker equations makes it possible to know the parameters of the filter and the variance of the white noise.

We can reformulate this matrix in the following form (We will use it in Laboratory Work):

$$\begin{bmatrix} R_{yy}(0) & R_{yy}(1) & \dots & R_{yy}(P-1) \\ R_{yy}(1) & R_{yy}(0) & \dots & R_{yy}(P-2) \\ \vdots & \vdots & \ddots & \vdots \\ R_{yy}(P-1) & R_{yy}(P-2) & \dots & R_{yy}(0) \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_P \end{bmatrix} = \begin{bmatrix} -R_{yy}(1) \\ -R_{yy}(2) \\ \vdots \\ -R_{yy}(P) \end{bmatrix}$$

$$\sigma^2 = R_{yy}(0) + \sum_{i=1}^P a_i R_{yy}(i)$$

Remarks: We do not have a random process but a single realization, i.e. $y(n)$, it is therefore not possible to calculate the statistical auto-correlation $R_{yy}(k)$. The latter will be replaced by temporal autocorrelation by assuming that the process is ergodic (see exo 1 of LW n°4).

There are various algorithms (Burg, Levinson) which make it possible to estimate the a_i fairly quickly and σ without going through matrix inversion. Just as it is possible to determine the adequate order N (AIC criterion).

Application example

1. We consider the auto-regressive (AR) model of order 1 such that: $x(n) = -a_1 \cdot x(n-1) + b(n)$

- Determine the Yule-Walker equations for this model
- Assuming $x(n)$ is known, determine the model parameters.

- Determine the $R_x(k)$ (the a_i are assumed to be known)

Answers:

- $(a_1 = -R_x(1)/R_x(0) \quad \sigma^2 = R_x(0)(1-a_1^2))$
- $(R_x(0) = \sigma^2 / (1-a_1^2) \quad R_x(1) = -a_1 \sigma^2 / (1-a_1^2) \quad R_x(k) = (-a_1)^k \sigma^2 / (1-a_1^2))$

2. Redo the same work for a model of order 2 such that: $x(n) = -a_1 x(n-1) - a_2 x(n-2) + b(n)$

Answers:

- $a_1 = R_x(1)[R_x(2) - R_x(0)] / [R_x(0)^2 - R_x(1)^2] \quad a_2 = [R_x(1)^2 - R_x(0)R_x(2)] / [R_x(0)^2 - R_x(1)^2]$
- $\sigma^2 = R_x(0) + R_x(1)^2 [R_x(2) - R_x(0)] / [R_x(0)^2 - R_x(1)^2] + R_x(2)[R_x(1)^2 - R_x(0)R_x(2)] / [R_x(0)^2 - R_x(1)^2]$
- $R_x(1) = -a_1 R_x(0) / (a_2 + 1) \quad R_x(2) = (-a_2 + a_1^2 / (1 + a_2)) R_x(0) \quad R_x(0) = (1 + a_2) \sigma^2 / (1 + a_2 - a_1^2 - a_1^2 - a_2^3 + a_2 a_1^2)$

-Levinson's algorithm

It is an algorithm which makes it possible to solve any system of the type $Ax = b$ with A Toeplitz, therefore in particular the normal equations of Yule-Walker:

$$\begin{bmatrix} R_{yy}(0) & R_{yy}(1) & \dots & R_{yy}(N) \\ R_{yy}(1) & R_{yy}(0) & \dots & R_{yy}(N-1) \\ & & \dots & \\ R_{yy}(N) & R_{yy}(N-1) & \dots & R_{yy}(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} \sigma^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

The principle of Levinson's algorithm is to recursively find the parameters of order k as a function of the parameters of order $k-1$.

- Initialization: $a_1(1) = k(1) = -\frac{R_{yy}(1)}{R_{yy}(0)}$

$$\sigma_1^2 = (1 - |a_1|^2) R_{yy}(0)$$

- Recursion: for k going from 2 to P (order of the AR model)

$$a_k(k) = -\frac{R_{yy}(k) + \sum_{i=1}^{k-1} a_{k-1}(i) R_{yy}(k-i)}{\sigma_{k-1}^2} = k(k)$$

$$a_k(i) = a_{k-1}(i) + a_k(k) a_{k-1}(k-i) \quad \text{pour } i = 1, \dots, k-1$$

$$\sigma_k^2 = (1 - |a_k(k)|^2) \sigma_{k-1}^2$$

3. Moving average (MA) model

The moving average signals are obtained by passing white noise through a purely transverse filter.

This filter is also called a finite impulse response filter: $H(z) = \sum_{i=0}^M b_i z^{-i}$

The signal $y(n)$ is assumed to be able to be written as a linear combination of samples decorrelated between them, which can be formalized as a linear combination of samples of a white noise $x(n)$.

So we have : $y(n) = \sum_{i=0}^M b_i x(n-i)$

And $\mu_y = E\{y(n)\} = \sum_{i=0}^M b_i \mu_x = \mu_x \sum_{i=0}^M b_i = \mu_x \cdot H_f(0)$

We are looking for the filter parameters that generate $y(t)$ from $x(t)$, centered white noise:

$$R_{yy}(k) = E\{y(n)y(n-k)\} = E\left\{\sum_{i=0}^M b_i x(n-i) \cdot \sum_{j=0}^M b_j x(n-j-k)\right\}.$$

$$R_{yy}(k) = \sum_{i=0}^M b_i \cdot \sum_{j=0}^M b_j E\{x(n-i) \cdot x(n-j-k)\} = \sum_{i=0}^M b_i \cdot \sum_{j=0}^M b_j R_{xx}(j+k-i)$$

- If $j+k \neq i \Rightarrow R_{yy}(k) = 0$
- Otherwise $\Rightarrow R_{yy}(k) = \sigma^2 \sum_{j=0}^{M-k} b_{j+k} \cdot b_j$

The problem is nonlinear according to the coefficients, it requires a nonlinear programming algorithm to obtain b_i from the $R_{yy}(k)$. However, Durbin's algorithm makes it possible to approach the optimal solution with good results. The principle of this algorithm consists of identifying the MA model of order M with an AR model of order $N \gg M$ (TP n°4). Indeed, any MA model can be identified with an AR model of infinite order: $\sum_{i=0}^M b_i z^{-i} = 1 / \sum_{i=0}^{\infty} a_i z^{-i}$

Example

We consider the adjusted mean (MA) model:

A. of order 1 such that $x(n) = e(n) + b_1 \cdot e(n-1)$

- Calculate μ_x .
- Assuming that $x(n)$ is known, determine the model parameters.
- Knowing the model parameters, determine the $R_x(k)$

B. of order 2 such that: $x(n) = e(n) + b_1 \cdot e(n-1) + b_2 \cdot e(n-2)$

- Calculate μ_x .
- Knowing the model parameters, determine the $R_x(k)$ Answers

$$\mu_x = 0 \quad R_x(0) = (1 + b_1^2) \cdot \sigma^2 \quad R_x(1) = b_1 \cdot \sigma^2 \quad R_x(k) = 0 \text{ for } k \geq 2$$

$$b_1 = (R_x(0) \pm \sqrt{R_x(0)^2 - 4R_x(1)^2}) / 2R_x(1) \quad \sigma^2 = 2 / (R_x(0) \pm \sqrt{R_x(0)^2 - 4R_x(1)^2})$$

$$R_x(0) = (1 + b_1^2 + b_2^2) \cdot \sigma^2 \quad R_x(1) = (b_1 + b_1 b_2) \cdot \sigma^2 \quad R_x(2) = b_2 \cdot \sigma^2 \quad R_x(k) = 0 \text{ for } k \geq 3$$

Note: It is very important to note that we have only one realization of the random signal to be modeled $y(n)$, therefore the statistical auto-correlation $R_{yy}(k)$ is obtained from the auto-correlation temporal by considering the ergodic process.

4. ARMA model

ARMA signals are obtained by passing white noise through a recursive filter also called an infinite impulse response (IIR) filter. These signals are a combination of AR and MA signals. The filter transfer function has a numerator and a denominator:

$$H(z) = \frac{\sum_{i=0}^M b_i z^{-i}}{1 + \sum_{i=1}^N a_i z^{-i}}$$

$$R_{yy}(k) = -\sum_{i=1}^N a_i R_{yy}(k-i) + \sigma^2 \sum_{j=0}^{M-k} b_{j+k} \cdot b_j$$

It is a nonlinear equation in a_i and b_j .

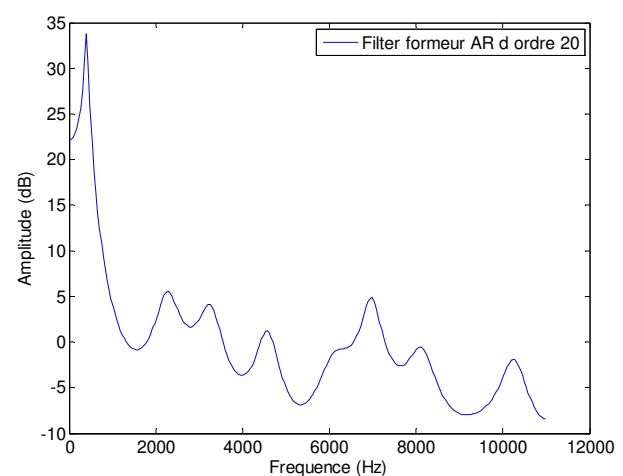
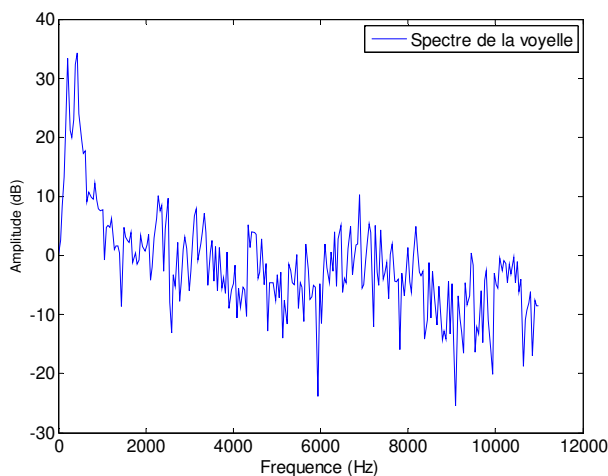
ARMA modeling can be broken down into AR modeling followed by MA modeling. The AR model presents a computational simplicity compared to the MA and ARMA models because the AR coefficients are solutions of a linear system of equations. Whereas the determination of the MA and ARMA coefficients requires the resolution of nonlinear equations. However, the ARMA model makes it possible to model both the minima and the maxima of the DSP and is therefore less restrictive than the AR model.

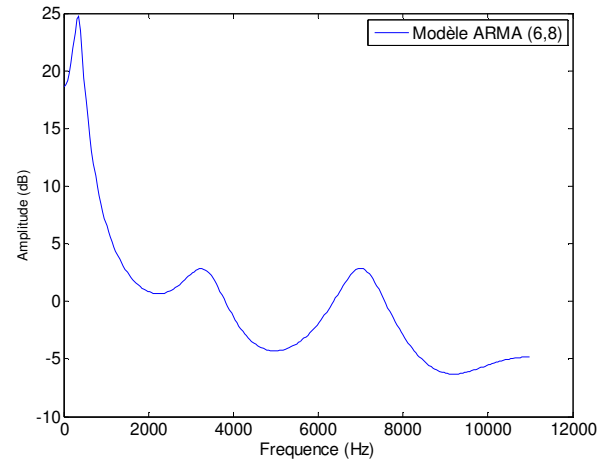
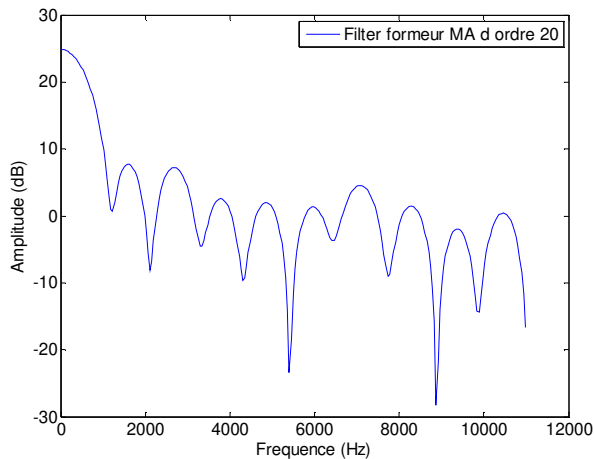
The applications of the AR, MA, ARMA models are numerous, among others:

- modeling and prediction of time series known as time series A time series is a sequence formed of observations over time that one seeks to model for the prediction of future data. Thus, in finance, this makes it possible to model the price of currencies or oil. While in meteorology, it allows to make forecasts on temperature or precipitation. In each case, we will try from a sample of data to build the best model that fits these data.

- estimation of the spectrum of a random signal, etc. This last application is based on the identification of the parameters of the model considered: The AR model is well suited to signals composed of pure lines in white noise. While the MA model is well suited for signals with zero power in certain frequency bands.

Example: Let's take the speech example again, for which we tested the three variants





Remember that for a low filter order, the model will not be able to capture all the relevant information of the signal, conversely if it is too high, it will model the noise. Remember that with the periodogram (fast algorithms: FFT) and its variants:

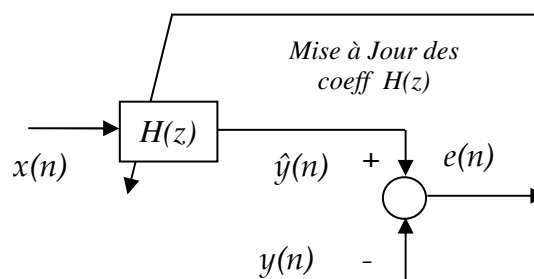
- Resolution in $f_e/N \Rightarrow$ Difficult to separate 2 close frequencies,
- Secondary lobes blocking frequencies at low amplitudes

We will not go any further in this course, the aim being only to introduce the notions of modeling and linear prediction.

6. Adaptive Digital Filtering

In Wiener filtering, the optimality criterion is stochastic: we want to minimize the mean squared error. This requires second-order process statistics (means and correlations). In practice, the correlation matrix $R_{xx}(k)$ and the cross-correlation $R_{yx}(k)$ are not known and must therefore be estimated from the observed data (temporal) assuming the ergodic processes.

The second problem is linked to the very hypothesis of stationarity which is not verified in the long term. Wiener filtering becomes inadequate for situations where the signal or noise is non-stationary. In such situations the optimal filter must be variable over time. The solution to this problem is provided by adaptive filtering.



Adaptive filtering involves a recursive update of the parameters (coefficients) of the filter. The algorithm starts from predetermined initial conditions and recursively modifies the filter coefficients to adapt to the

process. To do this, the coefficients of the filter's impulse response are adapted as a function of the error by a feedback loop.

- The gradient algorithm provides a recursive algorithm for calculating the coefficients of the Wiener filter (without going through the inversion of R_{xx}).
- The stochastic gradient algorithm (LMS) is a deterministic version in which the statistical quantities involved are replaced by instantaneous values.
- The least squares algorithm (RLS) which is a recursive algorithm of the LMS based on the minimization of the sum of squared errors over a given period of time.

6.1. Gradient (descent) algorithm

We seek to calculate the parameters of the optimal linear filter.

- At time n , these parameters are noted $h(n) = [b_0(n) \ b_1(n) \ \dots \ b_{N-1}(n)]^T$ and
- the inputs of the filter form the vector $X(n) = [x(n), x(n-1), \dots, x(n-N+1)]^T$.
- At each instant n , the output of the filter is a signal $\hat{y}(n)$ which one wishes to be as close as possible to the original signal $y(n)$.

We recall that the optimal solution given by the solution of the Wiener-Hopf equations is: $R_{xx}h = R_{yx}$ with $\xi(n) = E\{e^2(n)\} = E\left\{\left(y(n) - \sum_{i=0}^{N-1} b_i x(n-i)\right)^2\right\}$

The algorithm proceeds in three steps, an initialization step and two steps that are iterated:

1. Initialization: at time $n = 0$, we assign an initial value $h(0) = [0, 0, \dots, 0]$
2. At each instant n (the current instant), we calculate the gradient of the error (the derivative) $\nabla_{h(n)}\xi(n)$ with respect to the vector $h(n)$: $\nabla_{h(n)}\xi(n) = -2R_{yx} + 2R_{xx}h(n)$
3. Adjust the filter parameters according to: $h(n+1) = h(n) - \frac{\mu}{2}\nabla_{h(n)}\xi(n) = h(n) + \mu(R_{yx} - R_{xx}h(n))$

$\frac{\mu}{2}$: a small positive scalar which allows to adjust the speed and the precision of the adjustments. Note that there are versions with increasing adaptation μ_n step.

The idea of this algorithm is that adjustments in the opposite direction to the gradient (descent of the gradient) will drive $h(n)$ towards the value h_{opt} which is associated with the minimum error ξ_{min} . If the derivative is positive then we will decrease $h(n)$ so as to move towards the minimum of the cost function. The corrective term will therefore be chosen negative and vice versa

The algorithm is stable if $0 < \mu < \frac{2}{\lambda_{max}}$ where λ_{max} is the max of the eigenvalues of R_{xx}

This algorithm is also called deterministic gradient algorithm since the quantities R_{yx} and R_{xx} are perfectly known and determined. Also known as adaptive Wiener filtering.

6.1. Stochastic Gradient Algorithm (LMS)

The descent (gradient) method is an algorithm which estimates, by successive iterations, a solution which tends towards the optimal solution of the Wiener-Hopf equation, without inverting the matrix R_{xx} . Nevertheless, at each iteration, it is necessary to calculate the statistical correlations R_{xx} and R_{yx} which we have replaced by temporal correlations under the assumption of ergodicity.

The adaptive algorithm known as Least-Mean-Square (LMS) relies on simplifying the gradient of $\xi(n)$ either $\nabla_{h(n)}\xi(n) = -2R_{yx} + 2R_{xx}h$ by replacing statistical (ergodic: temporal) averages R_{xx} and R_{yx} instantaneous estimates:

$$\hat{R}_{xx} = X(n)X^T(n)$$

$$\hat{R}_{yx} = y(n)X(n)$$

So the gradient becomes $\hat{\nabla}_{h(n)}\xi(n) = -2y(n)X(n) + 2X(n)X^T(n)h(n)$

Note that minimizing the instantaneous squared error is less efficient and less precise than correcting the mean squared error.

$$\text{So, } \hat{h}(n+1) = \hat{h}(n) + \mu X(n) \overbrace{[y(n) - X^T(n)\hat{h}(n)]}^{e(n)}$$

$$\text{Either } \hat{h}(n+1) = \hat{h}(n) + \mu X(n)e(n)$$

1. Initialization: at time $n = 0, \hat{h}(0) = [0, 0, \dots \dots 0]$
2. Estimation of the error signal: $e(n) = y(n) - X^T(n)\hat{h}(n)$
3. Updating of the coefficients: $\hat{h}(n+1) = \hat{h}(n) + \mu X(n)e(n)$

The LMS algorithm converges on average, that is, for $n \rightarrow \infty, \hat{h}(n) \rightarrow h_{opt}$ of the Wiener-Hopf equations. Note that the gradient descent algorithm tends towards the minimum error ξ_{min} when n tends towards infinity and this in a regular exponential way, due to the exact estimation of R_{xx} and R_{yx} . On the contrary, the LMS algorithm, which relies on very rough approximations of its quantities, converges with chaotic oscillations around ξ_{min} , and this despite a correct choice of μ .

6.3. Least squares algorithm

The Wiener-Hopf equations are obtained by minimizing the mean square error $\xi(n) = E\{e^2(n)\} = E\{(y(n) - \hat{y}(n))^2\}$. The least squares algorithm is based on minimizing the sum of squared errors: $\xi(n) = \sum_{k=1}^n \lambda^{n-k} e^2(k)$ over a given period of time.

If $\lambda = 1$, $\xi(n)$ is sum of squared errors; If $\lambda < 1$, past errors are weighted with a (forgetting) factor which decreases exponentially.

Recursive version of RMS,

$$\xi(n) = \sum_{k=1}^n \lambda^{n-k} e^2(k) \quad \text{Ore}(k) = y(k) - \hat{y}(k) = y(k) - h^T(n)X(k)$$

$$R_{xx\lambda}(n)h(n) = R_{yx\lambda}(n)$$

$$\text{With } R_{xx\lambda}(n) = \sum_{k=1}^n \lambda^{n-k} X(k)X^T(k) = \lambda \sum_{k=1}^{n-1} \lambda^{n-k-1} X(k)X^T(k) = \lambda R_{xx\lambda}(n-1) + X(n)X^T(n)$$

$$\text{And } R_{yx\lambda}(n) = \sum_{k=1}^n \lambda^{n-k} y(k)X(k) = \lambda \sum_{k=1}^{n-1} \lambda^{n-k-1} y(k)X(k) + y(n)X(n) = \lambda R_{yx\lambda}(n-1) + y(n)X(n)$$

For $\lambda = 1$, the matrix $R_{xx\lambda}(n)$ is an approximation of the autocorrelation matrix. It is symmetrical such that: $R_{xx\lambda}(n)_{ij} = \frac{1}{n} \sum_{k=1}^n x(k-i+1)x(k-j+1)$

The algorithm looks like this:

$$\text{- Initialization } R_{xx\lambda}^{-1}(0) = \left(\sum_{k=-n_0}^0 X(k)X^T(k) \right)^{-1} \text{ and } R_{yx\lambda}(0) = \sum_{k=-n_0}^0 y(k)X(k)$$

$$-\hat{h}(n) = R_{xx\lambda}^{-1}(n)R_{yx\lambda}(n)$$

$$\text{with } R_{xx\lambda}(n) = \lambda R_{xx\lambda}(n-1) + X(n)X^T(n) \text{ and } R_{yx\lambda}(n) = \lambda R_{yx\lambda}(n-1) + y(n)X(n)$$

Note : Least squares filtering is deterministic. While the Wiener filter uses the method of least squares in stochastics.

Application examples

Adaptive filtering (in the sense of Wiener, in the sense of least squares, Kalman filtering, particle filtering) is used for various purposes including: modeling, prediction, system identification, interference cancellation (echo, noise), etc. allow the reconstruction of a signal in a noise environment

- Echo cancellation will provide us with a first example of using the LMS algorithm. Telephone (microphone and loudspeaker nearby).
- Cancellation of a jammer such as a signal disturbed by a sinusoidal jammer of known frequency but of which the amplitude A and the phase ϕ are unknown.

6.3. Other adaptive filters

- Kalman: The Kalman filter is used to estimate parameters of a system evolving over time from noisy measurements. It is particularly suitable for tracking because it not only allows parameters to be predicted but also measurement and model errors to be rectified. It operates similarly to adaptive filters. The prediction is made using the previous estimate. The error calculation then allows this prediction to be updated using the new measurements. Note that the extended version (Extended Kalman Filter) allows non-linear modeling to be taken into account.

- Particle (sequential Monte-Carlo method): An alternative to extended Kalman filters. The purpose of the filter is to estimate the posterior density of the variables that we seek to estimate (known as state variables) as a function of the observations (observation variables) assuming that the state variables constitute a Markov chain of first order.

It is used particularly for navigation applications: target tracking, missile guidance, robotics, etc.

FGE, USTHB [assiakourgli@gmail.com / <http://perso.usthb.dz/~akourgli/>]

Exercices n° 4: AR modeling and adaptive filtering

- Consider a random signal $y(n)$ SSL and ergodic whose temporal autocorrelation $\bar{R}_y(k) = \alpha^{|k|}$ with $0 < \alpha < 1$
 - Identify the appropriate linear model (AR or MA) for $y(n)$.
 - Assuming that the system $h(n)$ is purely recursive filter, give the schematic of the model by defining the input, the system, and the output.
 - Recall the necessary assumptions related to the use of this model.
 - We consider that the model is of order 1, determine its parameters.

- We consider the discrete-time linear filter defined by $y(n) = x(n) + b_1x(n-1) + b_2x(n-2)$.

where $X(n)$ and $Y(n)$ respectively designate the real random input and output processes of the filter where b_1 and b_2 are 2 real coefficients. We assume that $x(n)$ is a sequence of centered, independent random variables with variance σ^2 .

- Give the expression of $R_x(k)$ and $S_x(f)$ then give the expression of $R_y(k)$ and plot it for $b_1=1$ and $b_2=-1$.
- Knowing the DSP of the signal $y(n)$, what is the basis for the choice of the model?

- Consider an AR process defined by: $y(n) = -a_1y(n-1) - a_2y(n-2) + x(n)$ where $x(n)$ is decorrelated with variance 1

- Calculate $\mu_y(n)$ then without calculation, explain why $y(n)$ is SSL.
- Show that for $k > 0$, $R_y(k) = -a_1R_y(k-1) - a_2R_y(k-2)$
- Determine a_1 and a_2

- We want to model the signal $y(n)$ whose statistical autocorrelation is given in the figure opposite.

$R_y(k)$. It is assumed that the statistical autocorrelation of the input signal is $4\delta(k)$

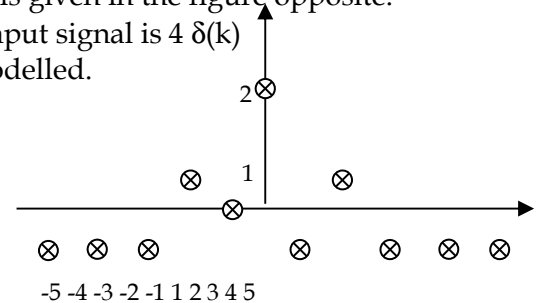
- Determine the first-order statistical moments of the signal to be modelled.

- Is this an AR or MA model? Justify

- Determine the order of the model

k

- Assuming that 2 coefficients are equal, from the values of $R_y(k)$, deduce that one of the coefficients is zero then determine the coefficients of this model then give the difference equation linking $y(n)$ and $x(n)$



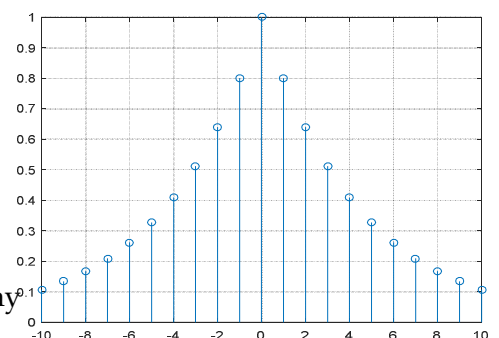
- We want to model the signal $y(n)$ SSL whose statistical correlation has only been plotted from -10 to 10. It is given in the figure opposite.

- Is this an AR or MA model? Justify

- Give the statistical properties of the input signal.

- We assume that the model is of order 1, determine its parameters.

- Explain the concept of shaping filter and its usefulness in telephony



Solutions

- AR model ($R_y(k) \neq 0$) $x(n)$ input is random signal to be modeled $h(n)$ shaping filter (math model) input is random + ergodicism $a_1 = -\alpha$, $\sigma_x^2 = 1 - \alpha^2$

- $R_x(k) = \sigma_x^2 \delta(k)$ $S_x(f) = \sigma_x^2$ MA of order 2 $R_y(0) = 1 + b_1^2 + b_2^2$ $R_y(1) = b_1(1 + b_2)$ $R_y(2) = b_2$ $R_y(k) = 0$ for $k \geq 0$

3. $\mu_y(n)=0$ input bb SSL \Rightarrow output SSL Interro1 14/15 4. See exam 17/18 5. Exam 16/17

Additional exercises

1. We consider a stationary random signal $x(n)$ and we assume its autocorrelation coefficients are known:

$$R(0) = 3\sigma^2, R(1) = 2\sigma^2, R(2) = \sigma^2, R(3) = 0$$

- We are looking for the MA filter of order 3 of this signal. Identify the filter settings.

- If the signal $x(n)$ was obtained by filtering a white Gaussian noise of variance σ^2 by a finite impulse response filter with transfer function $H(z) = 1 + az^{-1} + bz^{-2}$, deduce the values of a and b .

2. Consider a shaping filter whose difference equation is $y(n)=0.5(x(n)+x(n-1)+x(n-2)+x(n-3))$

- Explain the notion of shaping filter then identify this linear model AR or MA
- Give the mean, the autocorrelation and the PSD of its input $x(n)$.
- Calculate and plot $R_{yy}(k)$ then determine $S_{yy}(f)$

3. The signals $x(n)$ and $y(n)$ were obtained by filtering, by means of a finite impulse response filter, a centered Gaussian white noise $b(n)$ of variance σ^2 .

$$x(n) = 2b(n) + 0.5b(n-1) - 0.2b(n-2) + 0.1b(n-3) \quad y(n) = b(n) - b(n-2)$$

- Identify the 2 models then for each, calculate and plot the autocorrelation coefficients
- are $x(n)$ and $y(n)$ Gaussian (justify)
- Calculate the cross-correlations $R_{xy}(k)$, and $R_{yx}(k)$ and comment
- Give some applications of the AR, MA, ARMA models

4. Consider a shaper filter whose difference equation is $y(n) = -\alpha y(n-1) - \beta y(n-2) + x(n)$

- Identify the order of the AR linear model.
- Determine the parameters of the models, it is assumed that $R_{yy}(k) = 2 \cdot 0.5^{|k|}$

5. Consider a shaper filter whose difference equation is $y(n) = \alpha y(n-1) + x(n)$

- Identify the order of the AR linear model.
- Determine the mean of $y(n)$ and show that $R_{yy}(k) = \alpha^k R_{yy}(0)$, deduce a condition on α .
- Show that $R_{yy}(0) = R_{xx}(0) / (1 - \alpha^2)$
- Plot $R_{yy}(k)$ (Take $R_{xx}(0) = \sigma^2 = 1$).
- Name 2 concrete applications of AR models.

6. Consider a shaping filter whose difference equation is $y(n) = 0.25 y(n-1) - 0.25 y(n-2) + x(n)$

- Identify this AR or MA linear model (Justify)
- Determine the mean of $y(n)$ and give the expression for $R_{yy}(k)$.
- Compute and plot $R_{yy}(k)$ (Take $R_{xx}(0) = \sigma^2 = 1$).

Solutions

1. $b_0=1$ $b_1=1$ and $b_2=1$ and by identification, we find $a=1$ and $b=1$

2. MA, $\mu_x=0$ and $S_x(f) = \sigma^2$ $R_y(0) = \sigma^2$, $R_y(1)=R_y(-1)=0.75\sigma^2$, $R_y(-2)=R_y(2)=0.5\sigma^2$, $R_y(-3)=R_y(3)=0.25\sigma^2$, $R_y(k>3)=0$. $S_y(f) = \sigma^2(1 + 1.5\cos(4\pi f) + \cos(6\pi f) + 0.5\cos(8\pi f))$

3. Question 2 15/16 4. Ratt 15/16 5. Review 15/16 6. Ratt 14/15

Laboratory Work n° 5: Parametric modeling and adaptive digital filtering

This lab aims to:

- Model a random signal by AR and MA models (by AR approximation) and extract useful information.
- Test an adaptive filtering method (LMS)

Exercise 1: Download the file 'you have mail waiting.wav' and place it in the same directory as your program then select a part between 21000 and 21500

I. Demo

```
# -*- coding: utf-8 -*-
```

```
"""AR Modeling"""
```

```
# import numpy as np; import matplotlib.pyplot as plt;
# from scipy.io import wavfile as wf;
# import scipy. linalg as la; import scipy. signal as sp;

# fname = 'vousavezducourrierenattente.wav';
# #winsound.PlaySound(fname, winsound.SND_FILENAME)
# fe, S = wf.read(fname);
# Te=1/fe; N=len(S); t = np.linspace(0, N-1, N)*Te;
# plt. figure(1); plt. subplot(211); plt.plot(t,S); plt.grid(True);
# plt.xlabel('time'); plt.ylabel('Amp'); plt.title('Phrase');
# L=500;N1=21000;N2=N1+L; y=S[N1:N2]; ty=np.linspace(N1, N2-1, N2-N1)*Te;
# plt. subplot(212); plt. plot(ty,y); plt.grid(True);
# plt.xlabel('time'); plt.ylabel('Amp'); plt.title('part of the sentence');
```

```
# """ AR Modeling"""
```

```
# P=20; "P Number of coefficients of the AR model>2 "
# y=y-np.mean(y)
# R = np. correlate(y,y,mode='full')[len(y)-1:];
# #R=R/R.max(); "Autocor"
# C = la.toeplitz(R[0:P]);
# B = -R[1:P+1];
# a = la.inv(C).dot(B)
# a = np.append(1,a)
# sigma_square = sum(a*R[0:P+1]);
# """ Visualization of the frequency response of the filter found and the starting y signal"""
# b = np. array([1,0]);
# f,H= sp.freqz(b,a,L//2,fs=fe);
# TFy = np.fft.fft(y); TFy = TFy[1:L//2+1];
# plt.figure(2); plt.subplot(121);plt.plot(f, 20*np.log10(abs(TFy/TFy.max())));
# plt.plot(f, 20*np.log10(abs(H/H.max())));
# plt.title('Filter module found (orange)+ Original signal modeled (blue)');
# plt.xlabel('Freq ( Hz)'); plt.ylabel('Amp');
# # """ Synthesis from found filter"""
# bb = (sigma_square**0.5)*np.random.randn(L);
# #bb=bb-np.mean(bb);
# y_synt = sp.lfilter(b,a,bb);
```

```

# TFys = np.fft.fft(y_synt); TFys = TFys[0:L//2];
# plt.figure(2); plt.subplot(122); plt.plot(f, 20*np.log10(abs(TFys/TFys.max())));
# plt.plot(f, 20*np.log10(abs(TFy/TFy.max())));
# plt.title('synthesized signal (orange)+ synthesized signal (blue)');
# plt.xlabel('Freq ( Hz)'); plt.ylabel('Amp');
"""-----"""

""" LMS: Stochastic Gradient Algorithm: Restoration """
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sp

fecg=[]; meg=[];
""" Reading files """
with open('fecg.txt') as f:
    for i in f: fecg.append(float(i))
fecg = np.array(fecg)
with open('mecg.txt') as f:
    for i in f: mecg.append(float(i))
mecg = np.array(mecg)
""" Added noise """
N=len(mecg); var = 0.01;
bb=(var**0.5)*np.random.randn(N);
x=fecg+bb+mecg; #x=x-np.mean(x)
y=fecg; #y=y-np.mean(y)

""" Filter determination by LMS """
Ncoeff=7; mu=0.1;
h = np.zeros(Ncoef) # Initialization of the filter coefficients
L = len(x) # Signal length
yest = np.zeros(N) # y(n) estimated
En = np.zeros(N) # Error
for i in range(Ncoef,L//10):
    X= np. flipud(x[i-Ncoef+1:i+1]); #Take X=[x(n),x(n-1),...,x(n-Ncoef+1)]]
    yest[i] = np.dot(h, X)
    En[i] = x[i] - yest[i]
    h = h + mu * En[i] * X
h = h/sum(abs(h))

""" Viewing results"""
plt.figure(1)
y_est =sp.lfilter(h,1,x)
plt.subplot(311); plt.plot(x,'r', label= 'observation'); plt.plot(y,'b', label= 'fecg'); plt.grid(); plt.legend()
plt.subplot(313); plt.plot(En,'g',label='error'); plt.grid(); plt.legend()
plt.subplot(312); plt. plot(y,'b', label= 'fecg'); plt.plot(y_est,'r', label= 'ecg restored'); plt.legend()
"""-----"""

# """ Prediction by LMS """
# mux=0; varx=0.01; N=1000
# bb=mux+np.sqrt(varx)*np.random.randn(N)

# fe=10000; Te=1/fe;
# f0= 100;

```



```

# t = np.arange(0,N)*Te
# y = 0.5*np.cos(2*np.pi*f0*t)
# x = y + bb

# Ncoef=5; mu=0.2;
# hp = np.zeros(Ncoef) # Initialization of filter coefficients
# L = len(x) # Signal length
# yest = np.zeros(N) # y(n) estimated
# En = np.zeros(N) # Error
# for i in range(Ncoef,L-1):
# X= np. flipud(x[i-Ncoef+1:i+1]); #Take X=[x(n),x(n-1),...,x(n-Ncoef+1)]]
# yest[i] = np.dot(hp, X)
# En[i] = x[i] - yest[i]
# hp = hp + mu * En[i] * X

# plt.figure(2); plt.plot(x,'b', label= 'observation'); plt.plot(yest,'r', label= 'estimated')
# plt.plot(En,'g', label= 'Instant error'); plt.grid(); plt.legend()
"""-----"""

""" System identification by LMS """
# mux=0; varx=0.1; N=500
# x = mux+np.sqrt(varx)*np.random.randn(N)
# b = np.array([0.65, -0.35, 0.1])
# y = sp.lfilter(b,[1,0],x)

#Ncoef=3; mu=0.6;
# hs = np.zeros(Ncoef) # Initialization of filter coefficients
# yest = np.zeros(N) # y(n) estimated
# En = np.zeros(N) # Error
# for i in range(Ncoef,N-1):
# X= np. flipud(x[i-Ncoef+1:i+1]); #Take X=[x(n),x(n-1),...,x(n-Ncoef+1)]]
# yest[i] = np.dot(hs, X)
# ln[i] = y[i] - yest[i]
# hs = hs + mu * En[i] * X

# print(hs)
# plt. figure(3); plt.plot(y,'b', label= 'Output'); plt.plot(yest,'r', label= 'Estimated output')
# plt.plot(En,'g', label= 'Instant error'); plt.grid(); plt.legend()

"""To do AR"""

# import numpy as np; import scipy. signal as sp; import matplotlib.pyplot as plt
# import sounddevice as snd
# import scipy.io.wavfile as wav
# import scipy.linalg as la;

# def ModelAR(y,P):

```



```
# y=y-np.mean(y)
# R = np.correlate(y,y,mode='full')[len(y)-1:]
# C = la.toeplitz(R[0:P])
# B = -R[1:P+1]
# a = la.inv(C).dot(B)
# a = np.append(1,a)
# sigma_carre = sum(a*R[0:P+1]);
# return a,sigma_carre
```

```
# fe,x = wav.read('vowels.wav')
# snd. play(x, fe)
```

```
#1. Read, view and listen to the vowels.wav file
# 2. Identify the 3 parts corresponding to the 3 "A"
#NA=7000
#A1 = x[162400:162400+NA];
#A2 = x[253200:253200+NA]
#A3 = x[340000:340000+NA]
#3. Show the 3 "A's"
# 4. Calculate and display their periodogram in db
# 5. Determine the shaper filter of each "A" using the ModelAR function
# 6. Calculate the mean squared error (MSE) between the coefficients of each A
# from sklearn.metrics import mean_squared_error
# mseA12=mean_squared_error(a1,a2)
# 7. Repeat steps 2 to 6 for the letter "E"
# 8. Calculate the MSE between the 3 "A's" and the 3 "E's"
```

```
""""To do Restoration by LMS """"
""""
```

```
# 1. Play the audio you have mail waiting.wav in a variable y
# 2. Create a signal x composed of audio signal (y) + white noise
# 3. Listen to check for the presence of noise
# 4. Center the y and x variables
# 5. Determine by LMS the filter h by taking the part corresponding to silence (mu=0.01 and for i in
range(Ncoef,100) )
#6. Invert and normalize h-filter
# 7. Filter x by the filter
#8. Visualize Results
# 9. Listen to restored audio
# ss=input()
# snd.play(np.int8(y_est), fe)
""""
```

IV. Time-frequency and time-scale analysis

Most signals in the real world are not stationary, and it is precisely in the evolution of their characteristics (statistical, frequency, temporal, spatial) that most of the information they contain resides. Voice signals and images are common examples [27]. Remember that Fourier analysis allows a global characterization of the signal (we integrate from $-\infty$ to $+\infty$), we lose any temporal or spatial location, the ideal is to use a transformation which provides us with information on the frequency content while preserving the location (temporal or spatial) in order to obtain a time/frequency or space/scale representation of the signal. This is how two theories were developed, the window Fourier transform then the continuous wavelet transform.

Wavelets, a family of functions deduced from the same function, called the mother wavelet, by translation and dilation operations, have found, through the power of their theory, applications in numerous fields as varied as mathematics (analysis, probabilities, fractals), signal processing (compression, astronomy, seismic), physics (quantum mechanics, turbulence).

1. Time-frequency duality, short-term Fourier transform

The short-term Fourier transform (TFCT) and its derivatives (notably the spectrogram) are the time-frequency methods most used in practical applications. Thus, this class of methods represents the most widespread solution to eliminate the limitations of the Fourier transform. The basic idea is very simple and effective: we decompose the signal into small segments and we apply, on each of the sections, the Fourier transform, thus obtaining the “local” spectrum. The totality of the “local” spectra then indicates how the spectrum varies over time.

$$TFCT(t, f) = \int_{-\infty}^{+\infty} x(\tau) h^*(\tau - t) e^{-j2\pi f\tau} d\tau$$

Where $h(n)$ is a weighting window (Rectangular, Bartlett, Hanning, Hamming, Gaussian etc.)

The spectrogram is the squared module of the TFCT

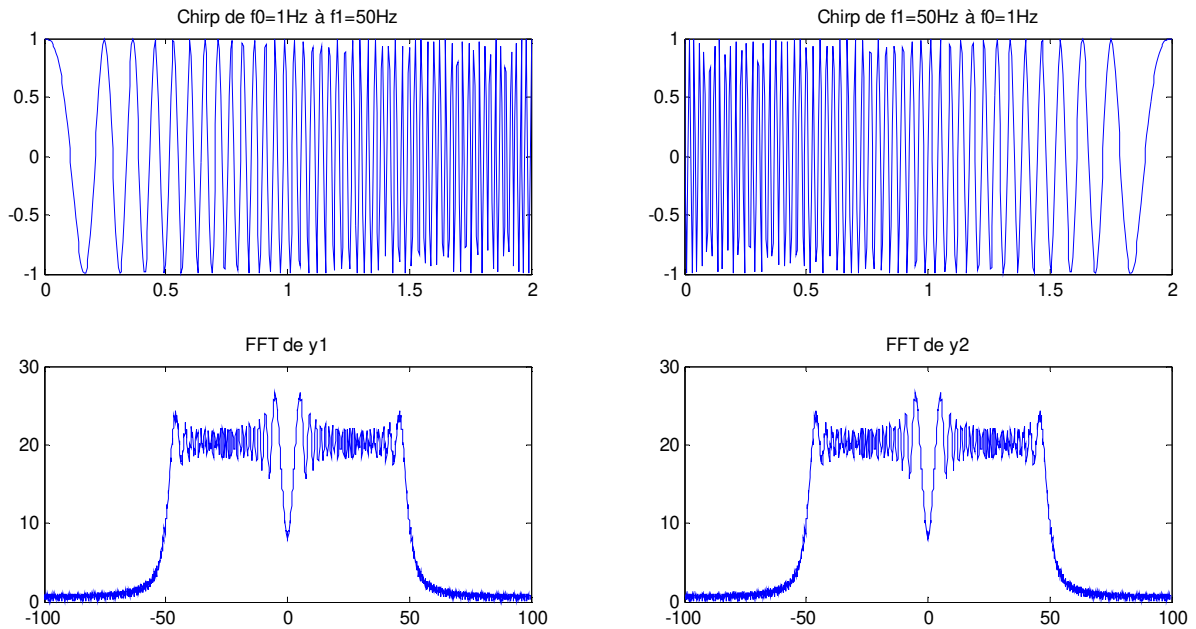
$$S_x(t, f) = \left| \int_{-\infty}^{+\infty} x(\tau) h^*(\tau - t) e^{-j2\pi f\tau} d\tau \right|^2$$

It should be noted that there is not just one TFCT since it depends on: The duration of the window (chosen so that the signal is assumed to be stationary over this duration), the shape of the window (width-height compromise of the lobes), the overlap rate between the windows.

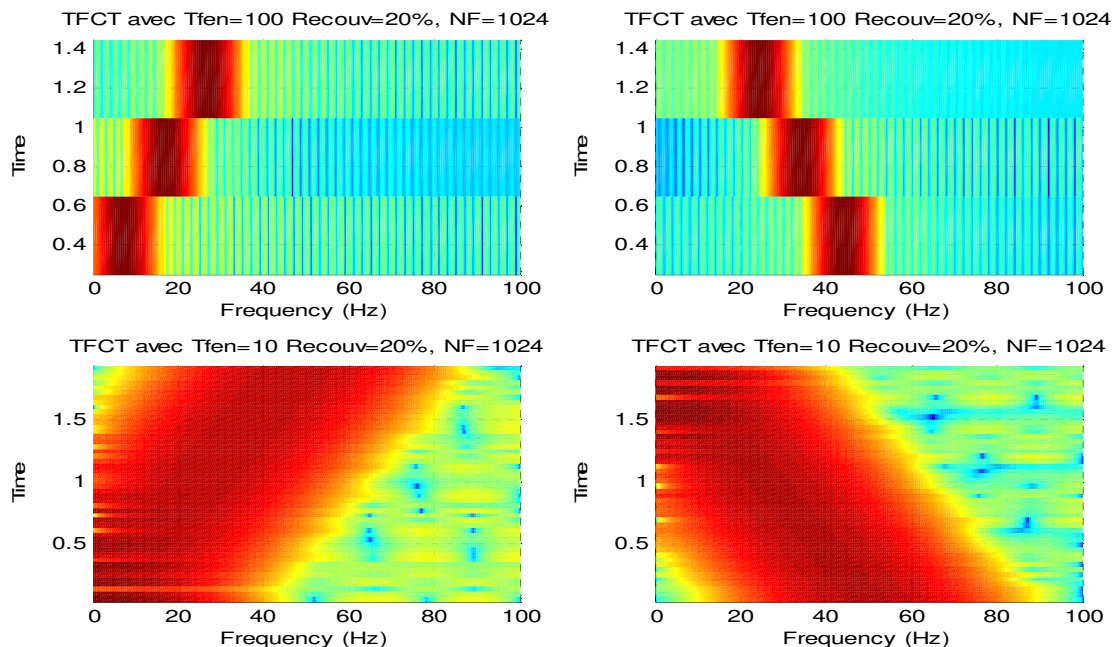
The role of the window $h(t)$ (whose energy must be equal to 1) is to cut out a neighborhood of length L of the point t , in which the frequency content is analyzed. We understand that there is a compromise between

the length L of $h(t)$ which represents the temporal resolution, which induces a frequency resolution in f_c/L , and the capacity of the TFCT to follow more or less rapid modulations. These 2 resolutions evolve inversely to each other. It has been shown (Heisenberg uncertainty principle) that the window which achieves the best time-frequency compromise is the Gaussian window.

Example: Analysis of a chirp signal going from f_1 then f_2 and vice versa (see Laboratory Work n° 6)



As illustrated above, we lose all temporal localization since we obtain the same TF for the two signals. To remedy this, the TFCT is calculated by performing the DFT for different intervals and overlaps



The TFDs on each LTling window provide the spectrogram which allows to adapt the TF to the characterization of non-stationary signals. One then obtains a time-frequency representation making it possible to locate the distribution of the energy simultaneously in time and frequency. Remember that the length of the chosen window will condition the number of frequency points (frequency resolution) and the

temporal resolution (average spectrum over the temporal window).

It is clear that for this case, the TFCT operating with a single window size does not make it possible to precisely locate each frequency. This should adapt according to the evolution of the signal. The ideal would be to be able to choose a window and a waveform (signal oscillating in a given time window) that one could expand (for low frequencies) and contract (high frequencies) at will.

2. Wigner-Ville transform (time-frequency)

Like the short-term Fourier transform, it makes it possible to illustrate the distribution of energy simultaneously in time and in frequency.

Recall that the spectral density is: $S_x(f) = |X(f)|^2 = TF\{R_x(\tau)\}$

$$\text{Or } R_x(\tau) = \int_{-\infty}^{\infty} x(t)x^*(t-\tau)dt = \int_{-\infty}^{\infty} x(t+\tau/2)x^*(t-\tau/2)dt$$

The expression $x(t+\tau/2)x^*(t-\tau/2)$ can then be considered as an instantaneous correlation, for which we can define an instantaneous spectral density more commonly called the Wigner-Ville time-frequency transform:

$$WV_x(t, f) = \int_{-\infty}^{\infty} x(t+\tau/2)x^*(t-\tau/2)e^{-2\pi jf\tau}d\tau$$

This transform retains many of the properties of the TF except that it does not preserve the linearity property :

$$x(t) = \sum_{k=1}^N x_k(t) \rightarrow WV_x(t, f) = \sum_{k=1}^N WV_{x_k}(t, f) + \sum_{k,m=1(k \neq m)}^N WV_{x_k x_m}(t, f)$$

The second term shows that this transform creates frequencies (interference) which have no real existence. In addition, it reveals negative energies causing difficulties of interpretation. A pseudo-smoothed version partially resolves some of these problems.

Examples :

$$-x(t) = e^{2\pi j f_0 t} \rightarrow WV_x(t, f) = \delta(f - f_0)$$

$$-y(t) = \cos(2\pi f_0 t) = \frac{e^{2\pi j f_0 t} + e^{-2\pi j f_0 t}}{2}$$

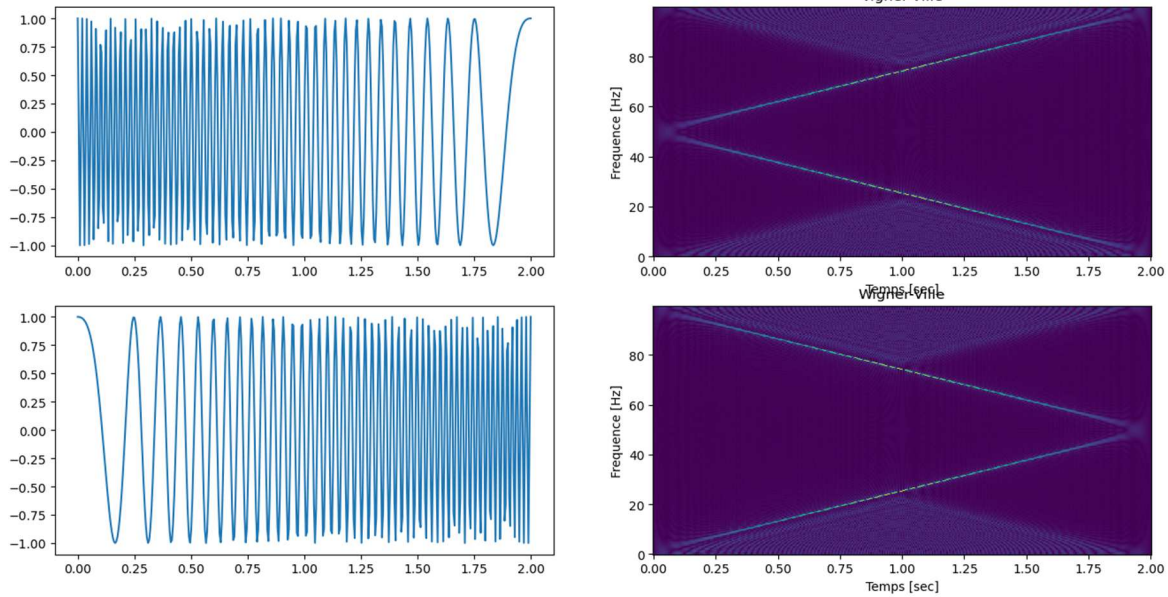
$$WV_y(t, f) = \frac{1}{4} \left(\int_{-\infty}^{\infty} [e^{2\pi j f_0 (t+\frac{\tau}{2})} + e^{-2\pi j f_0 (t+\frac{\tau}{2})}] [e^{-2\pi j f_0 (t-\frac{\tau}{2})} + e^{2\pi j f_0 (t-\frac{\tau}{2})}] e^{-2\pi j f \tau} d\tau \right)$$

$$WV_y(t, f) = \frac{1}{4} \left(\int_{-\infty}^{\infty} e^{2\pi j f_0 \tau} e^{-2\pi j f \tau} d\tau + \int_{-\infty}^{\infty} e^{4\pi j f_0 t} e^{-2\pi j f \tau} d\tau + \int_{-\infty}^{\infty} e^{-4\pi j f_0 t} e^{-2\pi j f \tau} d\tau + \int_{-\infty}^{\infty} e^{-2\pi j f_0 \tau} e^{-2\pi j f \tau} d\tau \right)$$

$$WV_y(t, f) = \frac{1}{4} (\delta(f - f_0) + e^{4\pi j f_0 t} \delta(f) + e^{-4\pi j f_0 t} \delta(f) + \delta(f - f_0))$$

$$\rightarrow WV_y(t, f) = \frac{1}{4} (\delta(f - f_0) + \delta(f + f_0)) + 2\cos(4\pi f_0 t) \delta(f)$$

Example : Analysis of a chirp signal going from f1 then f2 and vice versa



Despite the interference, good frequency localization is observed.

3. Continuous wavelets (time-scale)

It was introduced by Jean Morlet in 1981 to solve seismic signal problems in oil exploration.

Starting from a window h (known as mother function) whose symbol ψ depends on t , one can generate a set of similar basis functions by dilation (index a) and translation (index b) of a single prototype $\psi_{a,b}(t)$:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}} \psi\left(\frac{t-b}{a}\right) \quad a > 0$$

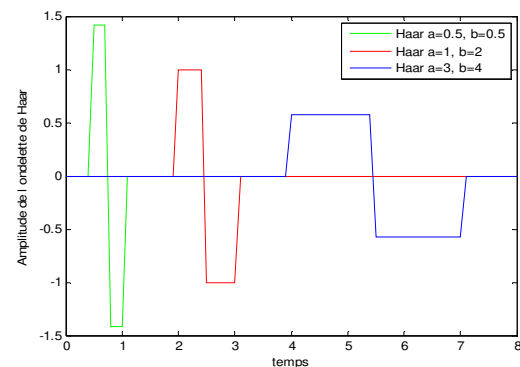
Where $a > 0$ is a contraction ($a < 1$) or expansion ($a > 1$) scale parameter of the window and b is a translation of the window. Note that the norm of $\psi_{a,b}$ is preserved when changing the scale factor (see figure below):

$$\|\psi_{a,b}\| = \int_{-\infty}^{+\infty} \frac{1}{a} \left| \psi\left(\frac{t-b}{a}\right) \right|^2 dt = \|\psi\|^2$$

Example :

$$\text{Haar's function} \quad \psi(t) = \begin{cases} 1 & \text{si } 0 \leq t \leq \frac{1}{2} \\ -1 & \text{si } \frac{1}{2} \leq t \leq 1 \\ 0 & \text{ailleurs} \end{cases}$$

$$\text{So } \psi_{a,b}(t) = \frac{1}{\sqrt{a}} \psi\left(\frac{t-b}{a}\right) = \begin{cases} \frac{1}{\sqrt{a}} & \text{si } b \leq t \leq b + \frac{a}{2} \\ -\frac{1}{\sqrt{a}} & \text{si } b + \frac{a}{2} \leq t \leq b + a \\ 0 & \text{ailleurs} \end{cases}$$



Wavelet Examples

- The Mexican hat wavelet (Paul to order 2) $\psi(t) = \frac{1}{\sigma\sqrt{2\pi}} \left(1 - \frac{t^2}{\sigma^2}\right) e^{-\frac{t^2}{2\sigma^2}}$

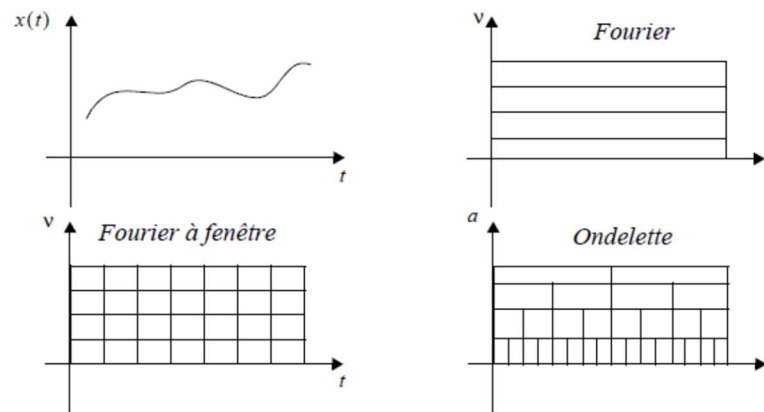
- Morlet's wavelet $\psi(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{t^2}{2\sigma^2}} e^{-2\pi jft}$

The Morlet wavelet is a complex sinusoid modulated by a Gaussian. The Paul wavelet decays more quickly than that of Morlet and allows more precise localizations in time. The wavelet is obtained by derivative of a Gaussian and allows temporal localizations of slightly lower quality than that of Paul [25].

By decomposing the signal $x(t)$ on this family, we thus obtain the wavelet coefficients $WT_{x,\psi}(a, b)$ which characterize the coefficient of the decomposition of the signal $x(t)$ in this basis, i.e.:

$$WT_{x,\psi}(a, b) = \int_{-\infty}^{\infty} x(t) \psi_{a,b}^*(t) dt = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} x(t) \psi\left(\frac{t-b}{a}\right)^* dt$$

The wavelet analysis starts with an analysis window a of very fine width, translates it over the entire signal then starts again by increasing the scale [25]. Its coefficients measure, in a certain sense, the fluctuations of the signal $x(t)$ around the point $t = b$, on the scale provided by a . By decreasing a , the support of $\psi_{a,b}$ reduces in time and therefore covers a larger frequency range and vice versa. So $1/a$ is proportional to a frequency.



For a large enough scale factor, the representation of the wavelet coefficients as a function of b , the position, gives a representation of “the general shape of the function”. On the other hand, a low scale factor corresponds to a representation of the singularities.

The wavelet transform is a linear operator, invariant by translation, and by dilation. Whatever the scale and whatever the place, the signal analysis is done with the same function. The wavelet transform of a signal is not unique, it depends on the mother wavelet used. Indeed, the mother wavelet $\psi(t)$ must have a good localization (null outside a certain interval), and must be oscillating (the number of null moments corresponds to the number of oscillations).

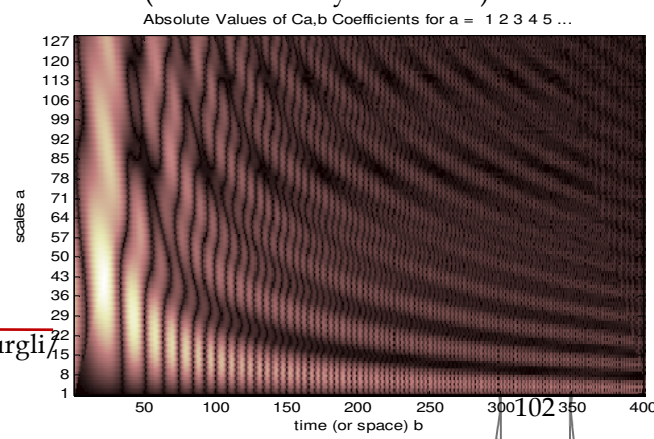
It can be shown that if the analyzing function (the wavelet) is correctly chosen, the wavelet transformation is invertible. The signal $x(t)$ can be reconstructed after double integration according to the scale factor a and the translation parameter b :

$$x(t) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \frac{1}{a^2} WT_{x,\psi}(a, b) \psi_{a,b}^*(t) da db$$

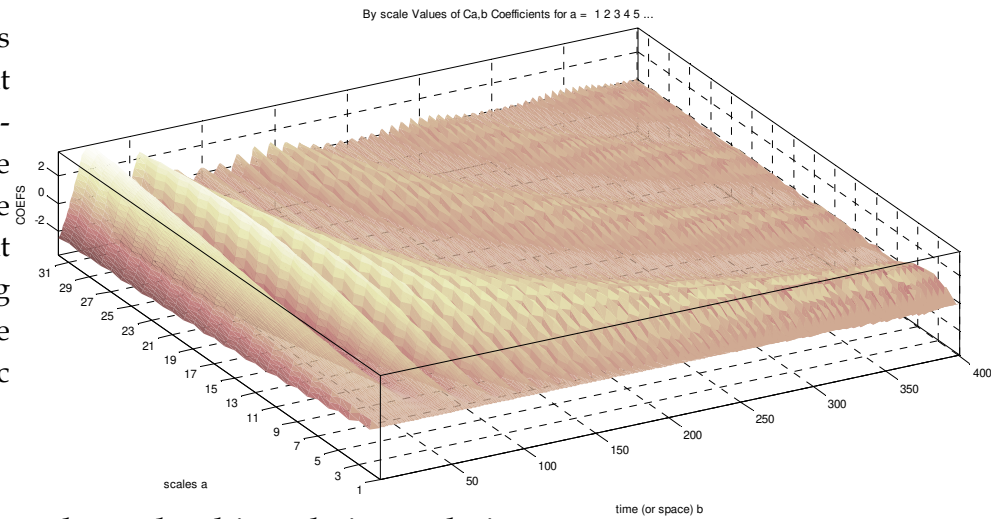
Example of application: Analysis of the chirp signal with the Haar wavelet (See Laboratory Work n°6)

The wavelet coefficients are represented as a function of time where the highest values are colored light. It is noted that a coefficient has an amplitude all the greater as the wavelet resembles the signal on the analyzed portion. When the window is narrow (narrow wave), the high frequencies are observed and when it is wide, it is the coefficients of the low frequencies which are high.

FGE, USTHB [assiakourgli@gmail.com / <http://perso.usthb.dz/~akourgli/>]



However, although the continuous wavelet transformation makes it possible to obtain a good time-frequency localization, however, the information generated by the transformation is infinitely redundant (a and b are continuous generating overanalysis). To remedy this, we resort to the use of discrete dyadic wavelets.



4. Discrete wavelets (DWT), dyadic wavelets and multi-resolution analysis

In applications, such as image processing, the amount of information to be processed can be significant, it is necessary to optimize the calculations and the size of the data. Multi-resolution provides a solution to this type of problem. This one is based on a discrete transformation which differs from the previous one by a factor of 2 (dyadic transformation). To limit the amount of information generated, the wavelet only encodes the difference in information between two successive resolutions. This technique lends itself well to a decomposition/composition by filter bank [25].

In most formulations of the DWT (discrete wavelet transform), the respective bases are therefore derived by a dyadic scale of the bases with $a=2^{-j}$ and a change of unit $b=k 2^{-j}$:

The discrete values of the scale factor a and of the translation parameter b will be considered in the form:

$$a = 2^{-j} \text{ et } b = k 2^{-j}$$

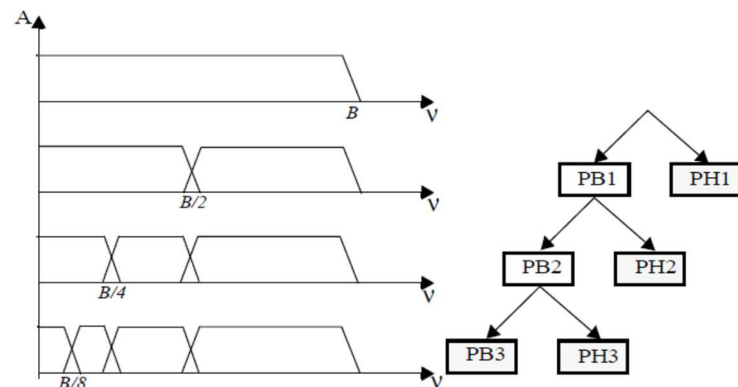
With these values of a and b , the equation becomes: $C_x(2^{-j}, k 2^{-j}) = 2^{\frac{j}{2}} \int_{-\infty}^{+\infty} x(t) \psi(2^j t - k) dt$

If the function $x(t)$ is discretized, assuming a sampling period equal to 1, for reasons of simplicity, the equation is then written: $C_x(2^{-j}, k 2^{-j}) = 2^{\frac{j}{2}} \sum_n x(n) \psi(2^j n - k)$

The construction of such wavelets can be approached as a problem of basic choice of signal decomposition, but also as a problem of decomposition into sub-bands (see previous chapter).

The idea of multiresolution analysis developed by Meyer and Mallat makes it possible to analyze a signal at different scales through linear operators at resolution levels corresponding to different spatial frequency bands.

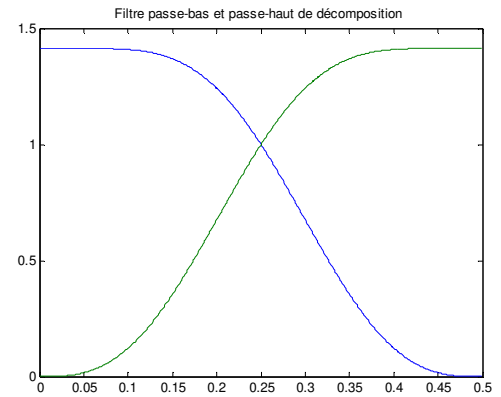
The signal is iteratively decomposed by a bank of filters. At each level, the signal spectrum is divided into two bands by a low-pass filter and one by a high-pass filter. The low pass filter (scaling function) gives the coarse information while the high pass filter (wavelets) encodes the details. The process is repeated until the desired resolution is achieved. The advantage of this



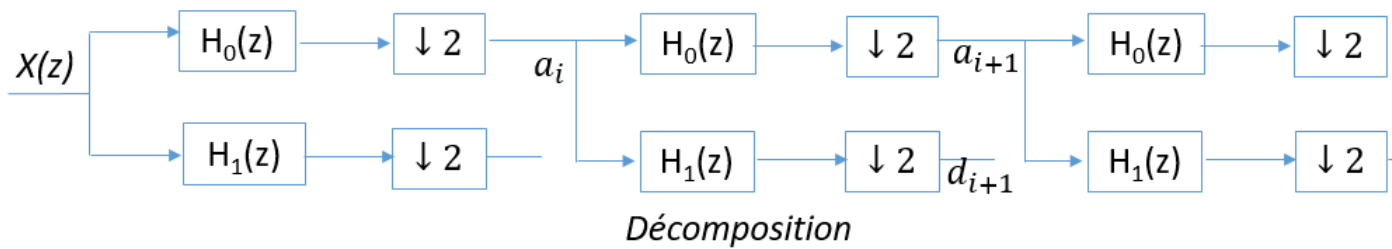
method is to use only two filters. When the process is stopped, the signal can be defined by the corresponding set of coefficients [25]: {PB3, PH3, PH2, PH1}.

The wavelet must be very compact so its coefficients must be, for the most part, close to zero. This condition depends on a few parameters such as the regularity of the function, the number of zero moments and the size of its support. The compactness of the support imposes that the filters (h_0) which will generate the scale function (giving a rough approximation of the signal) and the wavelet (h_1 providing the details) have a finite impulse response.

This condition requires the use of conjugated mirror filters. The two filters are symmetrically conjugated around the central frequency. (Low pass and high pass have the same cutoff frequency). The best known are the Daubechies filters with compact support. Quadrature filters have a finite impulse response. They use a low pass filter and a high pass filter. The term quadrature comes from the fact that the sum of the squared modules of the two filters is constant (2 or 1) known as the orthogonality condition. The bandwidth used by each of the filters is half of the initial band.). They intersect at $f_c/4$. In this case, we can use a decimator to reduce the quantity of information by a factor of 2 [25].



The discrete wavelet transform on orthonormal bases is reduced to digital filtering operations followed by undersampling. The reconstruction is perfect and is also carried out by digital filtering preceded by oversampling [27].



The decomposition following Mallat's algorithm is obtained [27]:

$$a_{j+1}[n] = \sum_k h_0[2n - k]a_j[k]$$

$$d_{j+1}[n] = \sum_k h_1[2n - k]a_j[k]$$

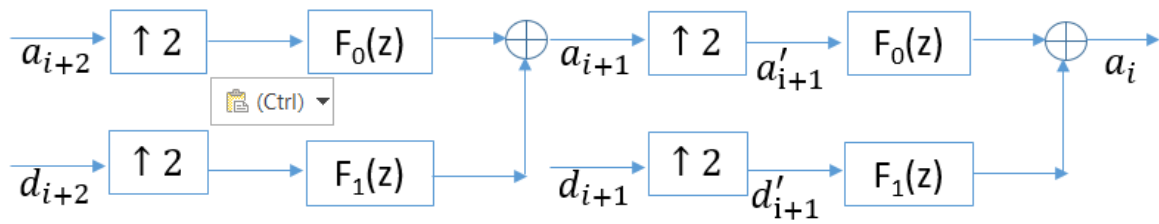
And the reconstruction following Mallat's algorithm is obtained by:

$$a_j[n] = \sum_k \{f_0[n - 2k]a_{j+1}[k] + f_1[n - 2k]d_{j+1}[k]\}$$

Noticed

If we call a'_{j+1} the signal a_{j+1} after interpolation (insertion of a 0 between the 2 successive samples) and d'_{j+1} the signal d_{j+1} after interpolation then [27]:

$$a_j[n] = \sum_k \{f_0[n - k]a'_{j+1}[k] + f_1[n - k]d'_{j+1}[k]\}$$



Reconstruction

A perfectly reconstructed filter bank can be obtained via CQF filters. To know the filters $h_0(n)$, $h_1(n)$, $f_0(n)$, et $f_1(n)$, in the case of orthogonal bases, it suffices to know only the filter $h_0(n)$ since the three other filters are calculated from the latter (reconstruction filters identical to the analysis filters but returned in time).

$$H_1(z) = H_0(-z^{-1})z^{-(L-1)} \quad \text{i.e. } h_1(n) = (-1)^n h_0(L-1-n)$$

$$F_0(z) = H_1(-z) = H_0(z^{-1})z^{-(L-1)} \quad \text{let } f_0(n) = h_0(L-1-n)$$

$$F_1(z) = -H_0(-z) \quad \text{i.e. } f_1(n) = (-1)^n h_0(n)$$

Example 1

$$H_0(z) = b_0 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3}$$

$$\Rightarrow H_1(z) = (b_0 - b_1 z^{-1} + b_2 z^{-2} - b_3 z^{-3})z^{-3} = -b_3 + b_2 z^{-1} - b_1 z^{-2} + b_0 z^{-3}$$

$$\Rightarrow F_0(z) = (b_0 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3})z^{-3} = b_3 + b_2 z^{-1} + b_1 z^{-2} + b_0 z^{-3}$$

$$\Rightarrow F_1(z) = -(b_0 - b_1 z^{-1} + b_2 z^{-2} - b_3 z^{-3}) = b_0 - b_1 z^{-1} + b_2 z^{-2} - b_3 z^{-3}$$

Thus, consider for example the 6-coefficient Daubechies wavelet such that the scaling function h either :

$$\checkmark \quad h_0 = [0.2352 \ 0.5706 \ 0.3252 \ -0.0955 \ -0.0604 \ 0.0249],$$

h_1 is obtained by going back h in time and reversing the sign of the odd coefficients, which gives us:

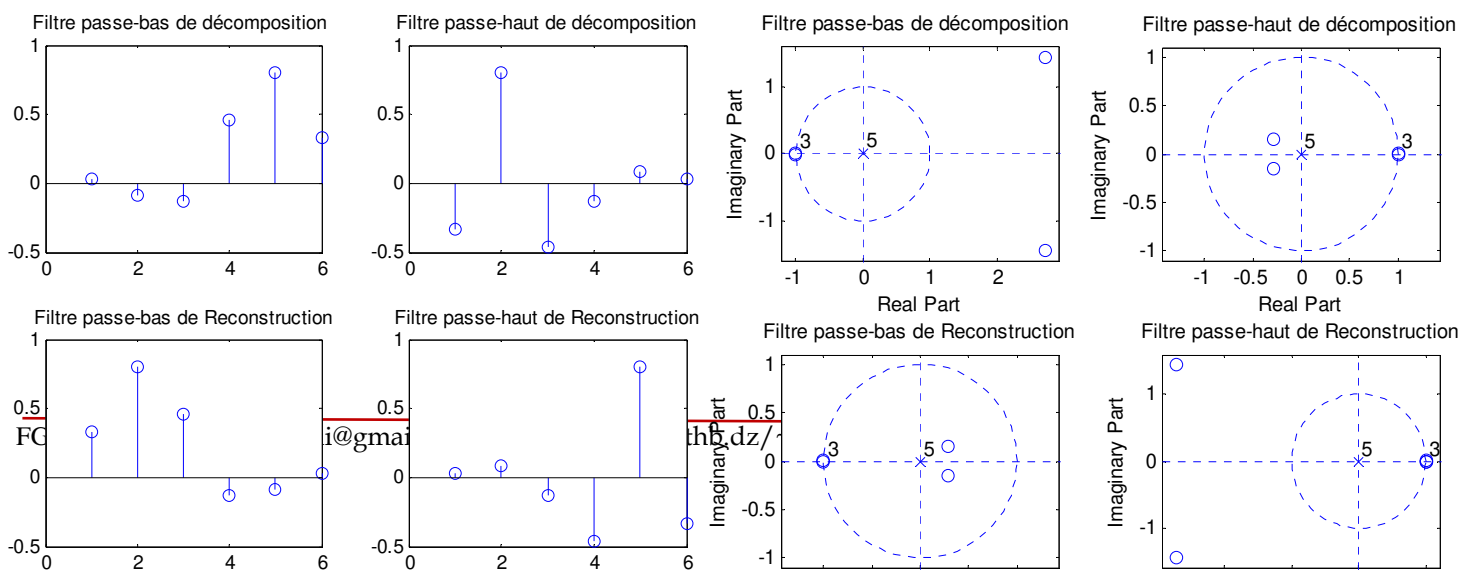
$$\checkmark \quad h_1 = [-0.0249 \ -0.0604 \ 0.0955 \ 0.3252 \ -0.5706 \ 0.2352]$$

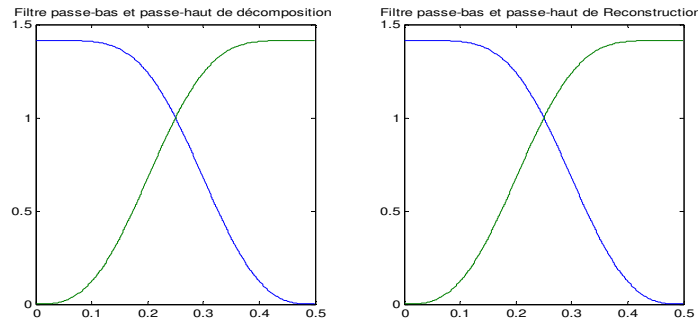
Reconstruction filters identical to analysis filters but flipped over time):

$$\checkmark \quad f_0 = [0.0249 \ -0.0604 \ -0.0955 \ 0.3252 \ 0.5706 \ 0.2352],$$

$$\checkmark \quad f_1 = [0.2352 \ -0.5706 \ 0.3252 \ 0.0955 \ -0.0604 \ -0.0249]$$

Below are given the wavelet coefficients of Daubechies 3 with the plot of the poles and zeros as well as the frequency responses (See LW n°6)





Note that these are CQF filters and therefore are orthogonal [28]:

$$|H_0(f)|^2 + |H_0(f \pm f_e/2)|^2 = 2, \quad |H_1(f)|^2 + |H_1(f \pm f_e/2)|^2 = 2$$

$$|H_0(f)|^2 + |H_1(f)|^2 = 2 \quad H_0(f)H_1^*(f) + H_0(f \pm f_e/2)H_1^*(f \pm f_e/2) = 0$$

Since the coefficients are not symmetrical, the phase shift will not be linear. Recall that the only orthogonal wavelet that is symmetric is the Haar wavelet.

Example of application 1: It is assumed that the signal to be analyzed is a ramp and that the wavelet is that of Haar.

Decomposition \mathbf{h}_0 Filters = $\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$, $\mathbf{h}_1 = \begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$ **Reconstruction** \mathbf{f}_0 Filters = $\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$, $\mathbf{f}_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$

Decomposition

$$\begin{aligned} a_{j+1}[n] &= \sum_k h_0[2n - k]a_j[k] = \sum_k f_0[k - 2n]a_j[k] & d_{j+1}[n] \\ &= \sum_k h_1[2n - k]a_j[k] = \sum_k f_1[k - 2n]a_j[k] \end{aligned}$$

	Approximation $\mathbf{h}_0[-k] = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} [f_0 k]$	Detail $\mathbf{h}_1[-k] = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} [f_1 k]$
Level 0	0,1,2,3,4,5,6,7	
Level 1	$\frac{1}{\sqrt{2}}, \frac{5}{\sqrt{2}}, \frac{9}{\sqrt{2}}, \frac{13}{\sqrt{2}}$	$-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}$

Reconstruction:

$$a_j[n] = \sum_k \{f_0[n - k]a_{j+1}[k] + f_1[n - k]d_{j+1}[k]\} = \sum_k \{h_0[k - n]a_{j+1}[k] + h_1[k - n]d_{j+1}[k]\}$$

	Approximation $\mathbf{f}_0[-k] = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \mathbf{h}_0[k]$	Detail $\mathbf{f}_1[-k] = \begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \mathbf{h}_1[k]$
Level 1	$\frac{1}{\sqrt{2}}, \frac{5}{\sqrt{2}}, \frac{9}{\sqrt{2}}, \frac{13}{\sqrt{2}}$ $0, \frac{1}{\sqrt{2}}, 0, \frac{5}{\sqrt{2}}, 0, \frac{9}{\sqrt{2}}, 0, \frac{13}{\sqrt{2}}$ $\frac{1}{2}, \frac{1}{2}, \frac{5}{2}, \frac{5}{2}, \frac{9}{2}, \frac{9}{2}, \frac{13}{2}, \frac{13}{2}$	$-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}$ $0, -\frac{1}{\sqrt{2}}, 0, -\frac{1}{\sqrt{2}}, 0, -\frac{1}{\sqrt{2}}, 0, -\frac{1}{\sqrt{2}}$ $-\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, \frac{1}{2}$
Level 0	0,1,2,3,4,5,6,7	

Application example 2 :

We assume that the low-pass filtering h_0 is an averaging operator between 2 coefficients and that the high-pass h_1 is an operator that encodes the half-difference between 2 coefficients [course 2013] such that $h_0=\{0.5, 0.5\}$, $h_1=\{-0.5, 0.5\}$, $f_0=2*\{0.5, 0.5\}$, $f_1=2*\{0.5, -0.5\}$

Original signal: 11, 9, 5, 7

Decomposition

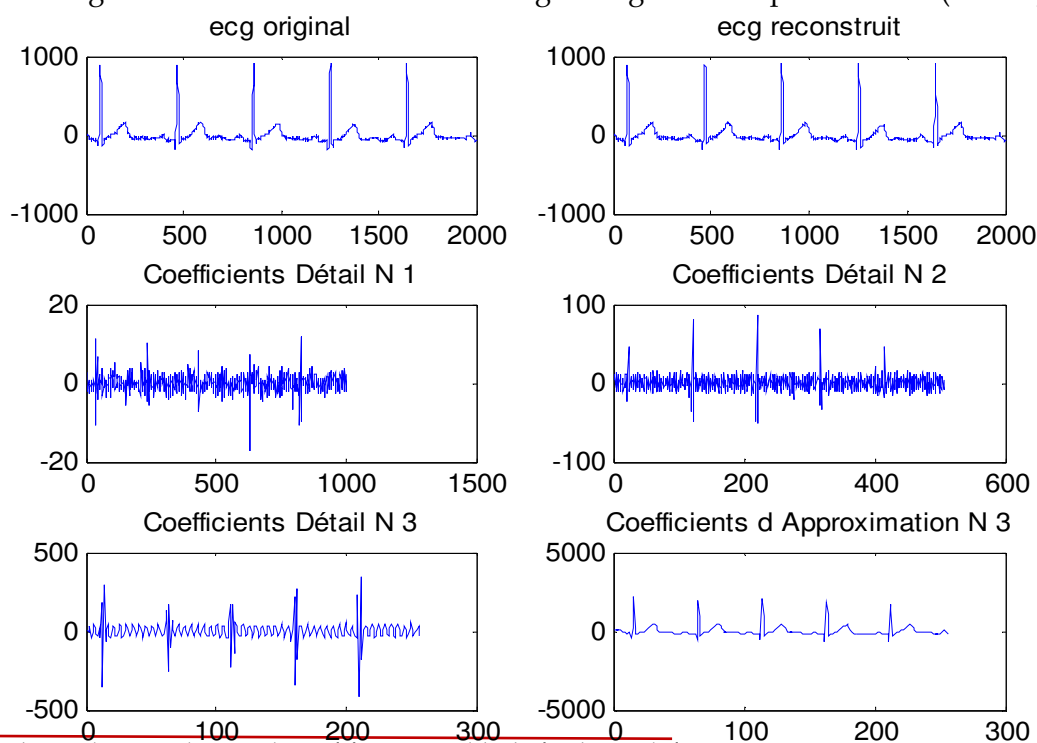
	Approximation $h_0[-k]=\{0.5, 0.5\}$	Detail $h_1[-k]=\{0.5, -0.5\}$
Level 1	10, 6	1, -1
Level 2	8	2
Decomposed signal	8, 2, 1, -1	

Reconstruction

	Approximation $f_0[-k]=\{1, 1\}$	Detail $f_1[-k]=\{-1, 1\}$
Level 2	8 0, 8, 0 [8, 8] +	2 0, 2, 0 [2, -2]
Level 1	10, 6 0.10, 0, 6, 0 [10, 10, 6, 6] +	1, -1 0, 1, 0, -1 [1, -1, -1, 1]
Signal reconstructed	11, 9, 5, 7	

The operator h_0 gives an approximate representation while h_1 gives a more precise representation (detail). At the end of the process, we keep the details and the last approximation, starting with the last level, we obtain: $S_D = \{8, 2, 1, -1\}$. It is possible to reconstruct the signal from the elements of the decomposition by summing the result of the application of filters f_0 and f_1 .

Example 3 Here is the result of the decomposition of the ecg signal, only the last approximation and all the levels of detail using which we can reconstruct the original signal are kept at the end (TP n°6)



Remarks :

- Daubechies filters lead to finite impulse response (FIR) filters but are not linear in phase.
- It is possible to construct wavelet bases leading to linear in-phase FIR filters such as B splines. These are the biorthogonal bases where families of decomposition and reconstruction are different and orthogonal only to each other [29]:

$$H_0(f)F_0^*(f) + H_0(f + f_e/2)F_0^*(f + f_e/2) = 2 \text{ And } H_1(f)F_1^*(f) + H_1(f + f_e/2)F_1^*(f + f_e/2) = 2$$

$$H_0(f)F_1^*(f) + H_0(f + f_e/2)F_1^*(f + f_e/2) = 0 \text{ And } H_1(f)F_0^*(f) + H_1(f + f_e/2)F_0^*(f + f_e/2) = 0$$

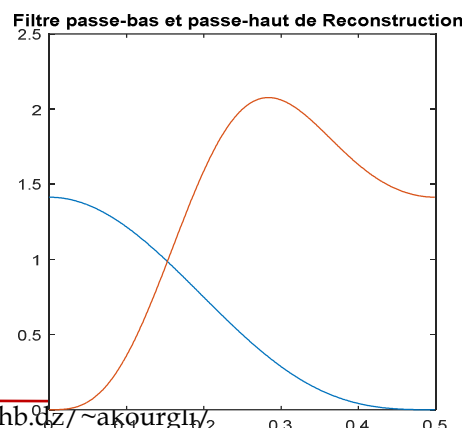
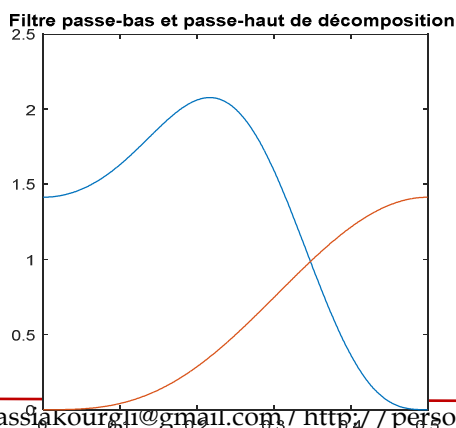
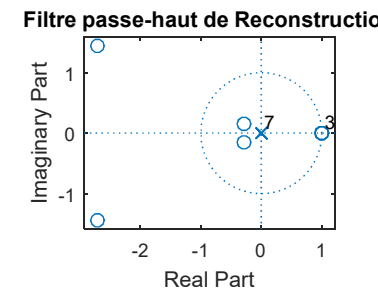
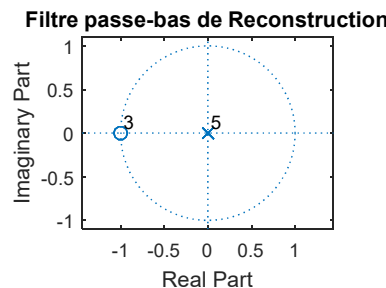
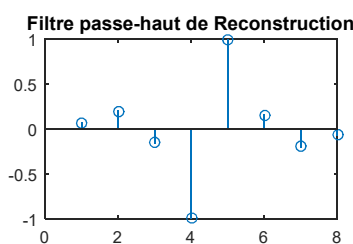
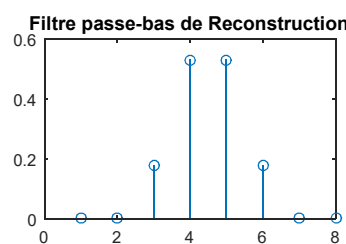
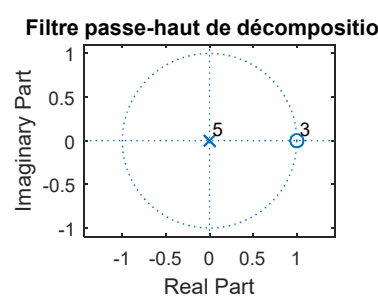
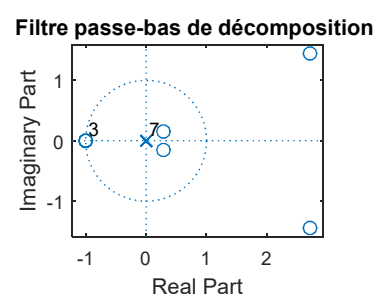
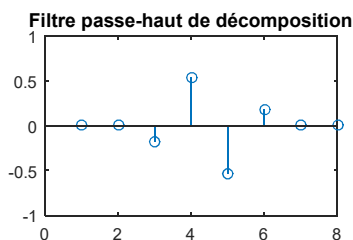
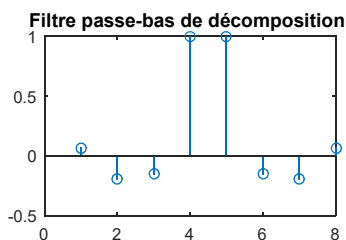
- In image processing, bi-orthogonal filters (JPEG2000) are preferred.

Example of biorthogonal wavelets

Low pass and high pass decomposition filters are different as are reconstruction filters. However, the decomposition and reconstruction filters are dual. Indeed, we obtain the high-pass reconstruction filter $F_1(z)$ by reversing the low-pass decomposition $H_0(z)$ in time and inverting the sign of every other coefficient. Likewise for the low-pass reconstruction filter $F_0(z)$ by reversing the low-pass decomposition $H_1(z)$ in time and inverting the sign of every other coefficient, namely:

$$h_1(n) = -(-1)^n f_0(L-1-n)$$

$$f_1(n) = (-1)^{nh_0}(L-1-n)$$



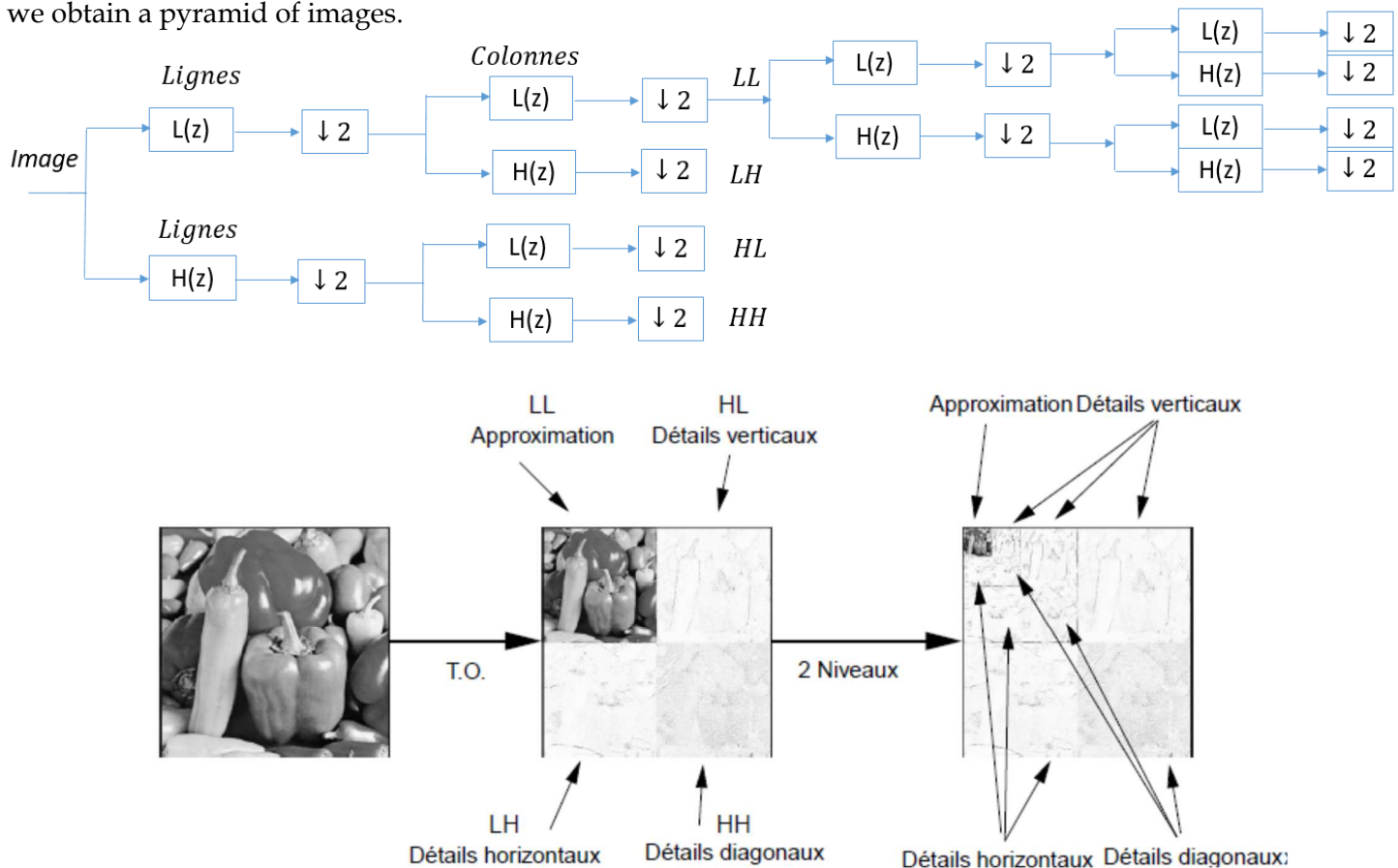
Note the symmetry of the impulse responses of the low-pass filters and the anti-symmetry of the high-pass analysis and reconstruction filters leading to a linear phase shift. Note also the duality between the analysis and reconstruction filters.

Note: The lifting scheme allows a very simple implementation of wavelet decompositions and their inverse operations using polyphase factorization [21]

5. Examples of application of the DWT on an image

Wavelets are well suited for signal denoising. The principle is simple, the noisy signal is broken down and the coefficients which do not pass a threshold are forced to zero. Then the signal is reconstructed (See Laboratory Work n°6). In the same way we can compress the signal.

For an image, we will apply two filters along the lines, one high pass (H), the other low pass (L), then, we only consider one column out of two and we apply the 2 again filters. Considering only one line out of two, we obtain 4 images reduced in size by half. By repeating this process on the approximation image (LL), we obtain a pyramid of images.



This example gives the result of jpeg and jpeg2000 compression (Wavelets) with a bpp of 0.2



Exercises n°4: Time-Frequency and Time-Scale Transforms

1. Determine the TFCT of the following signal (h is a gate window of width T):

$$x(t) = \begin{cases} e^{2\pi j f_1 t} & \text{pour } t < t_0 \\ e^{2\pi j f_2 t} & \text{pour } t > t_0 \end{cases}$$

2. Calculate the Wigner-Ville transform of the signal $x(t) = A_1 e^{2\pi j f_1 t} + A_2 e^{2\pi j f_2 t}$

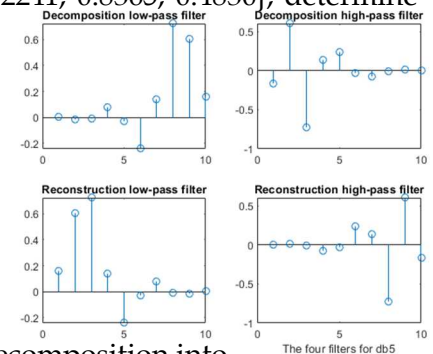
3. Consider the following wavelets:

- Haar's function $\psi(t) = \begin{cases} 1 & \text{si } 0 \leq t \leq \frac{1}{2} \\ -1 & \text{si } \frac{1}{2} \leq t \leq 1 \\ 0 & \text{ailleurs} \end{cases}$

- The Mexican hat wavelet $\psi(t) = \alpha(1 - t^2)e^{-\frac{t^2}{2}}$

Calculate their TF and plot the

4. Given the following low-pass decomposition filter: $h_0 = \{-0.1294, 0.2241, 0.8365, 0.4830\}$, determine the high-pass decomposition filter and the reconstruction filters



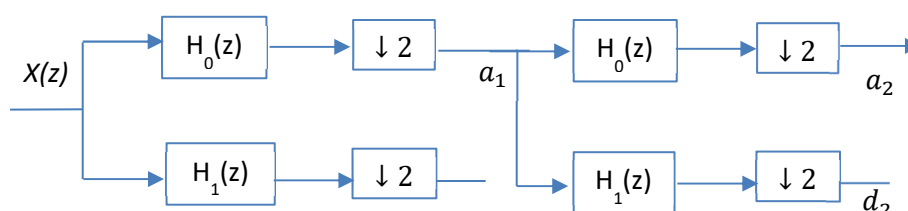
5. Let the decomposition and reconstruction filters be following, check the properties linking the 4 filters

6. Let the following signal $x = \{2, 4, 8, 12, 14, 0, 2, 1\}$ give and trace its decomposition into wavelets if the scale function is $h_0 = [0.5, 0.5]$

7. Consider the following signal: $f = \{4, 6, 10, 12, 8, 6, 5, 5\}$

- Give and plot its decomposition at level 1 into wavelets using the Haar wavelet.
- Reconstruct the signal from level 1
- Calculate its energy and then that after decomposition.
- Check that it is conserved and that 98% of this energy is in the approximated signal.
- Give the decomposition at levels 2 then 3.

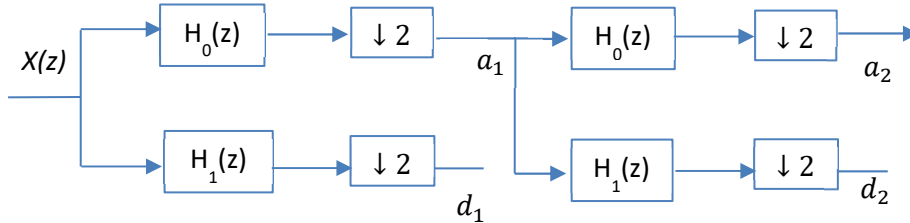
8. A signal $x(n)$ is decomposed by the Haar wavelet $h_0 = \{\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\}$ according to the diagram below. The first 8 values of the signal $x(n)$ are $\{4, 4, 6, 8, 8, 10, 9, 11\}$



d_1

- Determine a_1, d_1, a_2, d_2 , (**Detailed calculation**) then deduce the decomposed signal.
- Assuming that $f_e = 8\text{kHz}$, give the frequency bands occupied by a_1, d_1, a_2, d_2 ,
- Give the reconstruction scheme and the expression of filters $F_0(z)$ and $F_1(z)$.

9. We have decomposed a signal $x(n)$ by the Haar wavelet $h_0 = \{\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\}$ according to the diagram below . The first 8 values of the **decomposed signal** are $=\{4, 4, -1, 1, 2/\sqrt{2}, -1/\sqrt{2}, 0, 1/\sqrt{2}\}$



- Give the reconstruction scheme and the expression of the filters $F_0(z)$ and $F_1(z)$.
- Assuming that $f_e = 18\text{kHz}$, give the frequency bands occupied by a_1, d_1, a_2, d_2 ,
- Identify a_1, d_1, a_2, d_2 then **reconstruct** the original signal

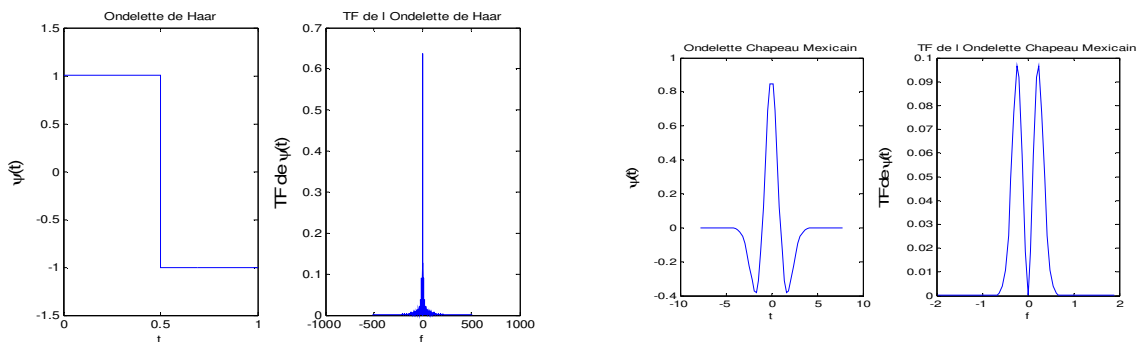
Solutions

1.

$$x(t) = \begin{cases} Tsinc((f-f_1)T)e^{-2\pi j(f-f_1)t} & \text{pour } t + \frac{T}{2} < t_0 \\ Tsinc((f-f_2)T)e^{-2\pi j(f-f_2)t} & \text{pour } t - \frac{T}{2} > t_0 \\ \left(t_0 - t + \frac{T}{2}\right) sinc\left(\left(t_0 - t + \frac{T}{2}\right)(f-f_1)T\right)e^{-2\pi j(f-f_1)\left(\frac{t_0+t}{2} + \frac{T}{4}\right)} + \\ \left(\frac{T}{2} - t_0 + t\right) sinc\left(\left(\frac{T}{2} - t_0 + t\right)(f-f_2)T\right)e^{-2\pi j(f-f_2)\left(\frac{t_0+t}{2} + \frac{T}{4}\right)} & \text{pour } -\frac{T}{2} < t - t_0 < \frac{T}{2} \end{cases}$$

$$2. WV_x(t, f) = |A_1|^2 \delta(f - f_1) + |A_2|^2 \delta(f - f_2) + 2A_1 A_2 \cos(2\pi(f_1 - f_2)) \delta\left(f - \frac{(f_1 + f_2)}{2}\right)$$

$$3. \text{Haar } \hat{\psi}(f) = j \frac{\sin^2\left(\frac{\pi f}{2}\right)}{\frac{\pi f}{2}} e^{-\pi j f} \quad \text{Mexican hat } \hat{\psi}(f) = \beta 4\pi^2 f^2 e^{-\frac{f^2}{2}}$$



$$4. h_1 = \{-0.4830, 0.8365, -0.2241, -0.1294\}, f_0 = \{0.4830, 0.8365, 0.2241, -0.1294\}, \\ f_1 = \{-0.1294, -0.2241, 0.8365, -0.4830\}$$

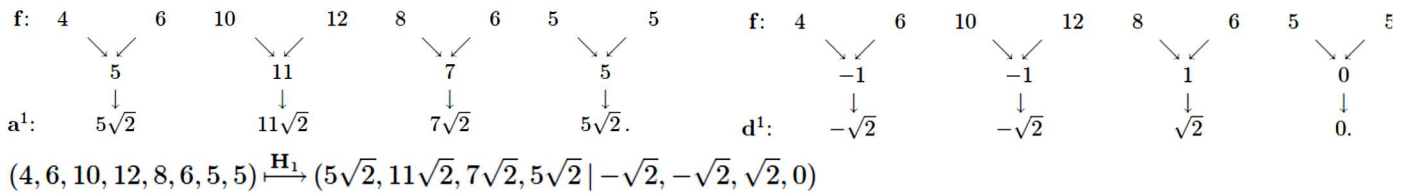
5.

Level	Approximation	Details
0	2 4 8 12 14 0 2 1	
1	3 10 7 1.5	-1 -2 7 0.5
2	6.5 4.25	-3.5 2.75

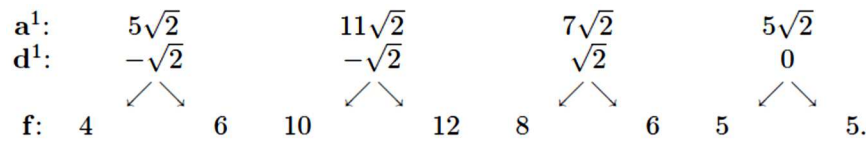
Decomposed signal: {5.375, 1.125, -3.5, 2.75, -1, -2, 7, 0.5}

7. Decomposition h_0 Filters = $\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$, $h_1 = \begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$ **Reconstruction** f_0 Filters = $\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$, $f_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$

Decomposition



Reconstruction

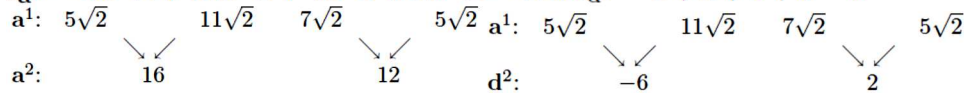


Energy

$$\mathcal{E}_f = 4^2 + 6^2 + \dots + 5^2 = 446.$$

$$\mathcal{E}_{(a^1 | d^1)} = 25 \cdot 2 + 121 \cdot 2 + \dots + 2 + 0 = 446$$

$$\mathcal{E}_{a^1} = 25 \cdot 2 + 121 \cdot 2 + 49 \cdot 2 + 25 \cdot 2 = 440 \quad \mathcal{E}_{d^1} = 2 + 2 + 2 + 0 = 6$$



$$(a^2 | d^2 | d^1) = (16, 12 | -6, 2 | -\sqrt{2}, -\sqrt{2}, \sqrt{2}, 0)$$

$$(a^3 | d^3 | d^2 | d^1) = (14\sqrt{2} | 2\sqrt{2} | -6, 2 | -\sqrt{2}, -\sqrt{2}, \sqrt{2}, 0)$$

88% of the energy is contained in a³ representing yet the 8thth of the signal

8. Review 19/20 9. Ratt 19/20

Additional Exercises

1. Wigner-Ville – Signal gaussian

$$z(t) = \frac{1}{\sqrt{\sigma}} e^{-\pi t^2 / \sigma^2} \longrightarrow W_z(t, f) = \sqrt{2} e^{-2\pi(t^2 / \sigma^2 + \sigma^2 f^2)}$$

– Signal monochromatique d'enveloppe gaussienne

$$z(t) = \frac{1}{\sqrt{\sigma}} e^{-\pi(t-t_0)^2 / \sigma^2} e^{2i\pi f_0 t} \longrightarrow W_z(t, f) = \sqrt{2} e^{-2\pi\alpha(t-t_0)^2 / \sigma^2 - 2\pi\sigma^2(f-f_0)^2}$$

– Chirp idéal

$$z(t) = e^{i\pi\beta t^2 + 2i\pi f_0 t} \longrightarrow W_z(t, f) = \delta(f - \beta t - f_0)$$

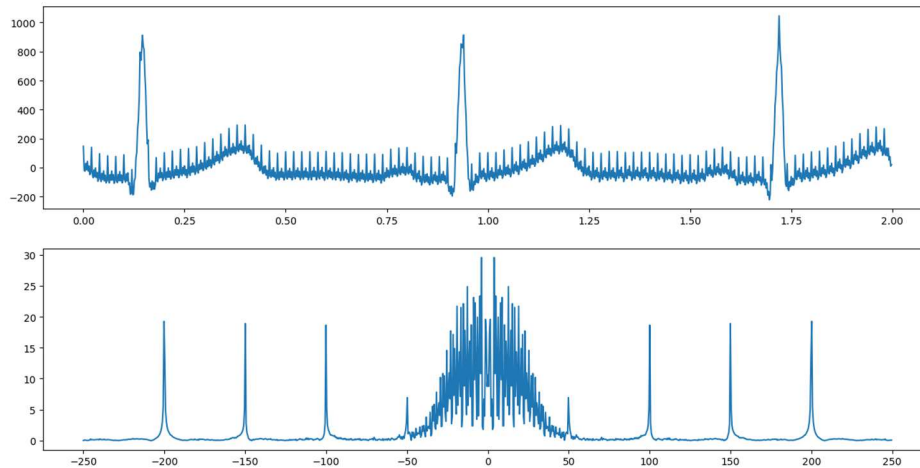
2. Consider the following signal: $x(n) = \{0, 1, -2, 3, -4, 5, -6, 7\}$

- Give its wavelet decomposition at level 1 using the Haar wavelet such that the decomposition low-pass filter is $h_0 = \{\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\}$
- Give the reconstruction diagram allowing to pass from one level to another.
- Rebuild the signal from level 1.
- Give the decomposed signal at levels 2 then 3.
- Check that the energy is conserved at all levels (1, 2 and 3)
- Propose a way to compress this signal.

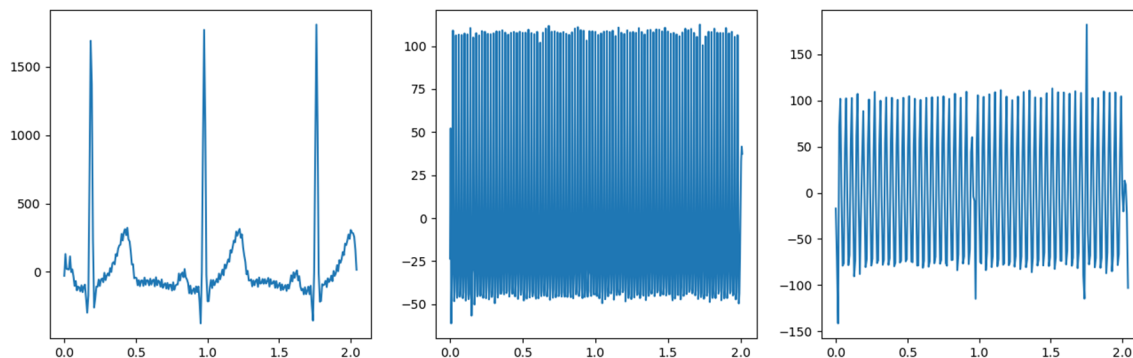
3. Consider the following signal: $x(n) = \{11, 9, 5, 7, 5, 11, 7, 9\}$

- Name a disadvantage of the continuous wavelet.
- Give the discrete wavelet decomposition scheme for 2 levels.

- Give the corresponding diagram for the frequency distribution.
 - Give its wavelet decomposition at level 2 using the following filters:
 $h_0=\{0.5, 0.5\}$, $h_1=\{-0.5, 0.5\}$, $f_0=2*\{0.5, 0.5\}$, $f_1=2*\{0.5, -0.5\}$
 - Is energy conserved?
 - Rebuild the signal from level 2.
4. During the transmission of an ecg signal, it was affected by a noise resulting from the combination of 3 sinusoids (see figure below: noisy signal + Modulus of its DFT).



By analyzing the signal by dyadic discrete wavelets, we obtained the following graphs:



- Identify for each of the 3 signals: the signal (approximate or detailed), the level of breakdown
- For each of the 3 signals, sketch the module of the DFT
- Propose a solution to remove the noise (detailed explanation)

Workarounds:

2. Exam 17/18

3. Ratt 17/18

4 . Exam 20/21

Laboratory Work n°6: From TFCT, Wigner-Ville to wavelets

This lab aims to:

- Become familiar with time-frequency and time-scale representations
- Approach the DWT, the notions of filter banks through the decomposition and synthesis filters
- Apply DWT to denoising

1. Demo

```
# -*- coding: utf-8 -*-
import numpy as np; import matplotlib.pyplot as plt; import scipy.signal as sp;
NF=1024; Te=0.005; fe=1/Te; N=round(2/Te);
t = np.linspace(0, 2, N);
x1= sp.chirp(t, f0=50, t1=2, f1=1, method='linear');
x2= sp.chirp(t, f0=1, t1=2, f1=50, method='linear');

""" TFD """
TFx1 = np.fft.fft(x1,NF); TFx1 = np.fft.fftshift(TFx1);
TFx2 = np.fft.fft(x2,NF); TFx2 = np.fft.fftshift(TFx2); freq = np.arange(-NF/2,NF/2)*fe/NF;
plt.figure(1)
plt.subplot(221); plt.plot(t, x1); plt.title('Chirp from 50 to 1Hz'); plt.subplot(222); plt.plot(freq, np.abs(TFx1));
plt.subplot(223); plt.plot(t, x2); plt.title('Chirp from 1 to 50 Hz'); plt.subplot(224); plt.plot(freq, np.abs(TFx2));

"""Spectrogram: TFCT"""
f, tt, Sxx1 = sp.spectrogram(x1, fe, nperseg=100, noverlap=20);
f, tt, Sxx2 = sp.spectrogram(x2, fe, nperseg=100, noverlap=20);
plt.figure(2)
plt.subplot(221); plt.plot(t, x1); plt.subplot(222); plt.pcolormesh(tt, f, Sxx1, shading='auto')
plt.ylabel('Frequency [Hz]'); plt.xlabel('Time [sec]'); plt.title('Spectrogram Tfen=100, TRec=20');
plt.subplot(223); plt.plot(t, x2); plt.subplot(224); plt.pcolormesh(tt, f, Sxx2, shading='auto')
plt.ylabel('Frequency [Hz]'); plt.xlabel('Time [sec]'); plt.title('Spectrogram Tfen=100, TRec=20');
f, tt, Sxx1 = sp.spectrogram(x1, fe, nperseg=20, noverlap=4);
f, tt, Sxx2 = sp.spectrogram(x2, fe, nperseg=20, noverlap=4);
plt.figure(3)
plt.subplot(221); plt.plot(t, x1); plt.subplot(222); plt.pcolormesh(tt, f, Sxx1, shading='auto')
plt.ylabel('Frequency [Hz]'); plt.xlabel('Time [sec]'); plt.title('Spectrogram Tfen=20, TRec=4');
plt.subplot(223); plt.plot(t, x2); plt.subplot(224); plt.pcolormesh(tt, f, Sxx2, shading='auto')
plt.ylabel('Frequency [Hz]'); plt.xlabel('Time [sec]'); plt.title('Spectrogram Tfen=20, TRec=4');

"""Wigner-Ville """
from tftb.processing import WignerCityDistribution
WVx1=WignerCityDistribution(x1,timestamps=t)
WVx2=WignerCityDistribution(x2,timestamps=t)
tfr_x1, t_x1, f_x1 = WVx1.run()
tfr_x2, t_x2, f_x2 = WVx2.run()
plt.figure(4)
plt.subplot(221); plt.plot(t, x1); plt.subplot(222); plt.pcolormesh(t_x1, f_x1*fe, np.abs(tfr_x1), shading='auto')
plt.ylabel('Frequency [Hz]'); plt.xlabel('Time [sec]'); plt.title('Wigner-City');
plt.subplot(223); plt.plot(t, x2); plt.subplot(224); plt.pcolormesh(t_x2, f_x2*fe, np.abs(tfr_x2), shading='auto')
plt.ylabel('Frequency [Hz]'); plt.xlabel('Time [sec]'); plt.title('Wigner-City');

"""Continuous wavelets"""
import pywt
Width = np.arange(1, 31);
cwtmatr,fs = pywt.cwt(x2, Width, 'mexh')
plt.figure(5)
plt.imshow(cwtmatr, aspect='auto', vmax=abs(cwtmatr).max(), vmin=-abs(cwtmatr).max())

""" Discrete Wavelets: Analysis Filters and Synthesis """
wavelet = pywt.Wavelet('haar')
```

```
plt.figure(6)
plt.subplot(221); plt.stem(wavelets.dec_lo); plt.title('Low Pass Decomposition');
plt.subplot(222); plt.stem(wavelets.dec_hi); plt.title('High Pass Decomposition');
plt.subplot(223); plt.stem(wavelets.rec_lo); plt.title('Reconstruction Low Pass');
plt.subplot(224); plt.stem(wavelets.rec_hi); plt.title('Reconstruction Low Pass');
L=512;
```

"""" Application of the DWT: Decomposition and Reconstruction """"

```
x = np.genfromtxt('ecg.dat');
coeffs = pywt.wavedec(x, 'db5', level=3)
cA3, cD3, cD2, cD1 = coeffs
plt.figure(7)
plt.subplot(321); plt.plot(x); plt.title('Original signal');
plt.subplot(322); plt.plot(cD1); plt.title('Level 1 detail');
plt.subplot(323); plt.plot(cD2); plt.title('Level 2 detail');
plt.subplot(324); plt.plot(cD3); plt.title('Level 3 detail');
plt.subplot(325); plt.plot(cA3); plt.title('Level 3 approximation');

x_reconst=pywt.waverec([cA3, cD3, cD2,], 'db5');
plt.subplot(326); plt.plot(x_reconst); plt.title('Signal Reconstructed');
```

2. To do

1. Set some level details to 0 and rebuild (remove on rebuild)
2. Choose one and set the detail values below the threshold to zero.
3. Test other wavelets and observe the quality of the reconstruction

1.1. Applying to an image

Using `pywt.dwt2`, decompose and reconstruct an image on several levels, set details to zeros and comment.

- F. Auger "Introduction to Signal and Information Theory" Technip
- G. Blanchet / M. Charbit "Digital signal processing: simulation using Matlab" Hermès
- M. Bouvet "Signal processing for sonar systems" Masson
- JM Brosier "Signal and digital communication" Hermès
- M. Charbit "Elements of signal theory: random aspects" Ellipses
- M. Gevers / L. Vandendorpe "Stochastic processes: estimation and prediction" UCL
- Mr. Kunt "Digital Signal Processing / TNS Workshop" Dunod / PPR
- Y. Thomas "Signals & linear systems: course / exercises" Masson
- F. Truchetet "Linear digital signal processing" Hermès