II

Traitement numérique des signaux

«In fact, any operation that can be completely described to the required accuracy (if numerical) in a finite number of steps using the words « if », « or », « and », etc, can be done automatically with relays.»

A Symbolic Analysis of Relay and Switching Circuits, 1937

CLAUDE ELWOOD SHANNON (1916-2001)

PLAN DU CHAPITRE

I L'échantillonnage

I.1 Principe

Tout traitement d'un signal par un système informatique nécessite dans un premier temps son acquisition sous forme numérique. Cette étape consiste à enregistrer les valeurs prises par le signal à intervalles de temps réguliers. Un telle technique porte le nom **d'échantillonnage.**
Chaque échantillon prélevé est immédiatement converti en valeur binaire par un **C**onvertisseur **A**nalogique-**N**umérique ou C.A.N. puis stocké en mémoire.

Le signal ainsi numérisé est enregistré sous forme de liste ou tableau de valeurs numériques, généralement des flottants, et accompagnés d'erreurs d'arrondis lié à la quantification du CAN (cf *quantification* plus bas dans ce cours).

QUESTION : comment fonctionne schématiquement la chaine d'acquisition échantillonneur-bloqueur, puis CAN ?

RÉPONSE : les valeurs de $f(t)$ sont d'abord prélevées tous les $T_e = \dfrac{1}{F_e}$ à l'aide d'un échantillonneur-bloqueur par capacité, avec T_e appelée période d'échantillonnage (F_e "fréquence d'échantillonnage"). On enregistre en mémoire la suite des nombres suivants : $[f(0), f(T_e), f(2T_e), \ldots, f(nT_e), \ldots]$ à l'aide du CAN (dont on ne détaillera pas le principe dans ce cours-> cf TD).

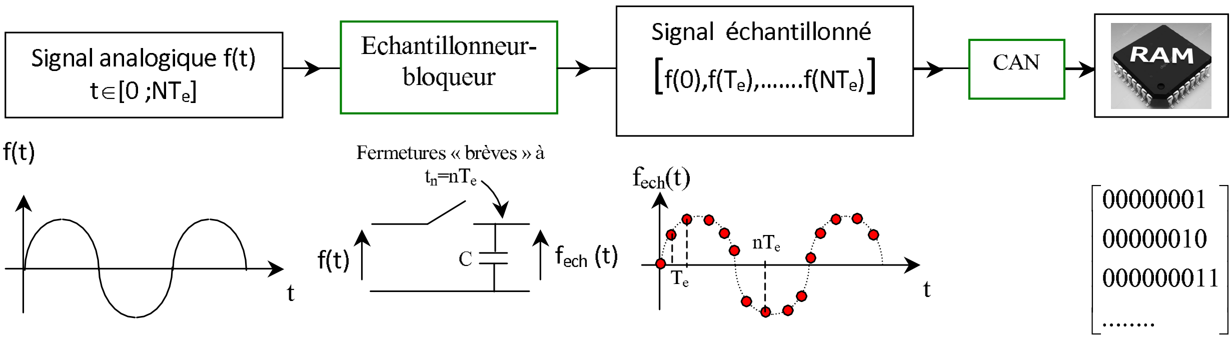

FIGURE II.1 – Principe de l'échantillonnage

Le signal échantillonné peut-être représenté par une fonction mathématique $f_{ech}(t)$.

ECRITURE FORMELLE DE $f_{ech}(t)$:

- **pour l'échantillon $f_{ech}(nT_e)$ pris seul à la date $t_n = nT_e$ la valeur est donnée par le produit suivant :**

$$f_{ech}(nT_e) = f(nT_e) \times \delta(t - nT_e) \qquad \text{avec } \delta(t-nT_e) \text{ pic de Dirac d'amplitude 1 centré sur la date } nT_e$$

$$= f(t) \times \delta(t - nT_e) \qquad \text{car } \delta(t-nT_e) = 0 \text{ pour } t \neq nT_e$$

- **pour un signal continu $f(t)$ ($t \in]-\infty; +\infty[$), la totalité du signal échantillonné $f_{ech}(t)$ est donnée par la sommation suivante :**

$$f_{ech}(t) = \sum_{n=-\infty}^{+\infty} f_{ech}(nTe) = \sum_{n=-\infty}^{+\infty} f(nT_e) \times \delta(t - nT_e)$$

$$= \sum_{n=-\infty}^{+\infty} f(t) \times \delta(t - nT_e)$$

soit finalement : $\boxed{f_{ech}(t) = f(t) \times \underbrace{\sum_{n=-\infty}^{+\infty} \delta(t - nT_e)}_{\text{peigne de Dirac période } T_e}}$

L'opération d'échantillonnage peut donc être résumée très schématiquement ainsi :

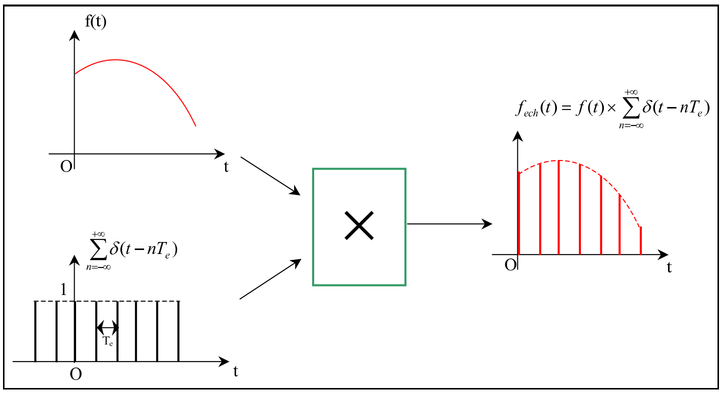

FIGURE II.2 – Schema de principe de l'échantillonnage

I.2 Analyse spectrale d'un signal échantillonné

a - Rappel : spectre d'un peigne de Dirac

Le peigne de Dirac correspond ici à un train d'impulsion de largeur tendant vers 0 ($\Delta t \to 0$) de période T_e (période d'échantillonnage) et d'amplitude E (on rappelle la contrainte $E \cdot \Delta t = 1$) :

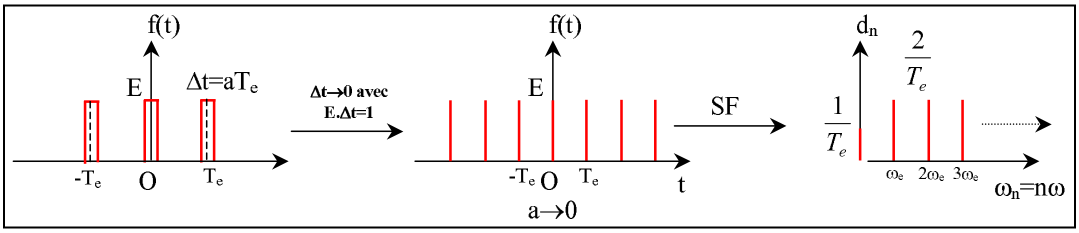

FIGURE II.3 – Série de Fourier d'un Peigne de Dirac ($a = \frac{\Delta t}{T_e} \to 0$)

On rappelle les coefficients de Fourier d'un tel signal : $\begin{cases} d_0 = |c_0| \overset{|sinc(na\pi)|\to 1\ \forall n}{\longrightarrow} Ea = E\dfrac{\Delta t}{T_e} = \dfrac{1}{T_e} \\ d_{n\geq 1} = |2\underline{c_n}| \overset{|sinc(na\pi)|\to 1\ \forall n}{\longrightarrow} 2E\dfrac{\Delta t}{T_e} = \dfrac{2}{T_e} \\ \varphi_n = arg\left(\underline{c_n}\right) = 0 \end{cases}$

Ainsi, la SF du peigne de Dirac de période T_e est : $\boxed{\mathscr{P}_{\mathscr{D}}(t) = E\cdot\sum_{n=-\infty}^{+\infty}\delta(t-nT_e) = \frac{1}{T_e} + \sum_{n=1}^{n=+\infty}\frac{2}{T_e}\cdot\cos(2\pi n\frac{t}{T_e})}$

b - Spectre d'un signal échantillonné $f_{ech}(t)$

- CAS D'UN SIGNAL SINUSOÏDAL PUR

  Considérons dans un premier temps un signal sinusoïdal pur de fréquence f_0 : $f(t) = A\cos(2\pi f_0 t)$ que l'on échantillonne à T_e. Le signal échantillonné s'écrit :

$$f_{ech}(t) = A\cos(2\pi f_0 t)\times[\text{Peigne Dirac}] = A\cos(2\pi f_0 t)\times\left[\frac{1}{T_e} + \sum_{n=1}^{n=+\infty}\frac{2}{T_e}\cdot\cos(2\pi nF_e t)\right]$$

soit : $$f_{ech}(t) = \frac{A}{T_e}\left[\overbrace{\cos(2\pi f_0 t) + \sum_{n=1}^{n=+\infty}\cos\left(2\pi(nF_e - f_0)t\right) + \cos\left(2\pi(nF_e + f_0)t\right)}^{\text{décomposition en somme d'harmoniques}}\right]$$

  Le spectre du signal est donc constitué d'une infinité de composantes toutes de même amplitude :

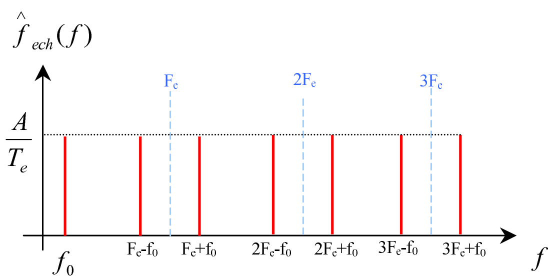

FIGURE II.4 – Spectre du signal sinusoïdal échantillonné

- CAS D'UN SIGNAL QUELCONQUE

  Considérons maintenant un signal quelconque $s(t)$ en particulier non périodique, donc de spectre $\hat{s}(f)$ continu. Ce dernier peut par exemple avoir l'allure suivante :

Remarque I-1: CHOIX DE F_{max}

NB : F_{max} représente la fréquence maximale présente dans le spectre du signal. Lorsqu'il s'agit d'un signal périodique, nous savons que l'étendue spectrale de celui-ci est infinie et que l'amplitude des harmoniques décroît avec la fréquence ; dans ce cas, on retient pour F_{max} la fréquence du dernier harmonique de poids significatif dans le spectre [a].

a. pour un signal créneau ($\sim \frac{1}{n}$), abandonner le rang 20 est suffisant ; pour un signal triangulaire ($\sim \frac{1}{n^2}$), ce sera le rang 5

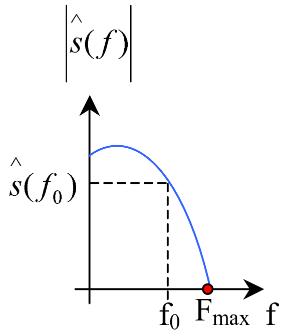

FIGURE II.5 – Spectre du signal $s(t)$ non périodique (seules les fréquences positive de la TF sont représentées)

On peut par exemple isoler la composante de fréquence f_0, d'amplitude $|\hat{s}(f_0)|$; d'après la démarche ci-dessus, le spectre de cette composante une fois échantillonnée contient les fréquences $nF_e \pm f_0$. En reproduisant ce raisonnement pour l'ensemble des composantes spectrales du signal $s(t)$, on obtient le spectre $\hat{s}_{ech}(f)$ représenté ci-dessous, qui reproduit donc le motif spectral tous les nF_e :

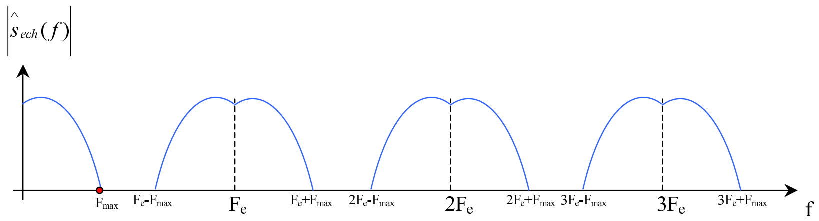

FIGURE II.6 – Spectre $\hat{s}_{ech}(f)$ du signal échantillonné $s_{ech}(t)$

c - Reconstitution d'un signal à partir de son échantillonnage (technique passe-bas) - théorème de Nyquist-Shannon

QUESTION : comment obtenir le spectre $\hat{s}(\omega)$ du signal $s(t)$ à partir de celui du signal échantillonné $\hat{s}_{ech}(\omega)$?

RÉPONSE : **en réalisant un filtrage passe-bas** du signal échantillonné :

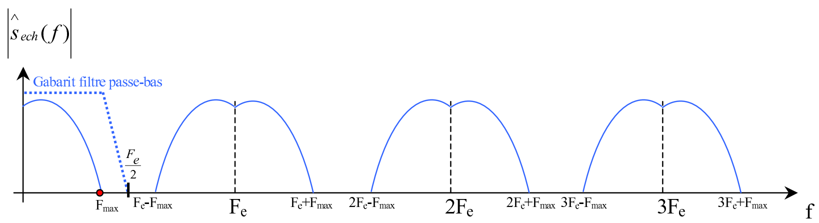

FIGURE II.7 – Filtrage passe-bas du signal échantillonné $s_{ech}(t)$

Pour restituer correctement le spectre du signal originel après filtrage, il faut s'assurer qu'aucune portion du "massif spectral" centré sur F_e ne pénètre dans le "massif spectral" centré sur 0 que l'on doit récupérer ; la condition de non chevauchement s'écrit naturellement :

$$F_{Max} < F_e - F_{Max} \qquad \text{soit} \qquad \boxed{2F_{Max} < F_e}$$

On en déduit le **théorème de Nyquist-Shannon imposant des conditions pour réaliser un échantillonnage correct** :

Propriété I-1: THÉORÈME DE NYQUIST-SHANNON

La reconstitution *parfaite*, i.e. sans perte d'information (par synthèse de Fourier), d'un signal à partir de son échantillonnage à la fréquence F_e n'est possible qu'à condition qu' F_e soit au minimum deux fois supérieure à la plus haute fréquence présente dans le spectre du signal, ou bien la plus haute fréquence que l'on retiendra comme significative dans le spectre :

$$\boxed{\text{Echantillonnage correct} \Leftrightarrow Fe > 2F_{Max} \quad \text{soit} \quad F_{Max} < \frac{F_e}{2} = F_{Ny}}$$

en posant $F_{Ny} = \dfrac{F_e}{2}$ **fréquence de Nyquist**.

NB : la fréquence de coupure du filtre est souvent fixée à $f_c = F_{Ny} = \dfrac{F_e}{2}$ (cf. fig II.10)

EXEMPLE : cas du CD-AUDIO

Pour un CD-AUDIO, on a $F_e = 44,1\ kHz \simeq 44\ kHz$ et le spectre des audio-fréquences $F \in [20\ Hz\ 20\ kHz]$; l'échantillonnage est donc correct pour une restitution ultérieure du signal :

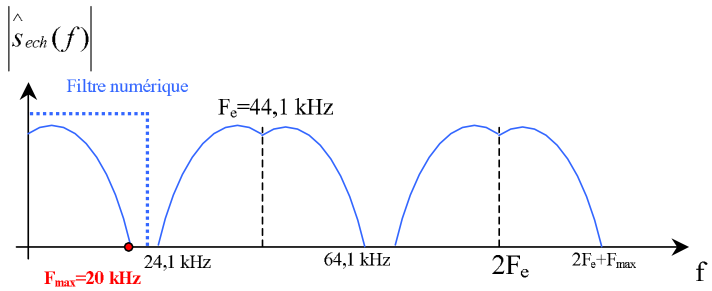

FIGURE II.8 – Echantillonnage à la norme CD-audio

d - Non respect du théorème de Nyquist-Shannon : le repliement de spectre et les fréquences "fantômes" (Expérience de cours/Simulation Python)

EXPÉRIENCE DE COURS : illustration du repliement sur un signal sinusoïdal $f_0 = 1\ kHz$.

Reprenons le cas simple de l'échantillonnage d'un signal sinusoïdal et fixons sa fréquence à $f_0 = 1kHz$.

Deux situations peuvent présenter :

- La fréquence d'échantillonnage est $F_e > 2F_{Max} = 2kHz$ par exemple $F_e = 3kHz \implies$ **l'échantillonnage est correct (condition de Nyquist-Shannon respectée)**

- La fréquence d'échantillonnage est $F_e < 2F_{Max} = 2kHz$ par exemple $F_e = 1{,}5kHz \implies$ **l'échantillonnage est incorrect ! (condition de Nyquist-Shannon violée)**

  Dans ce second cas, le spectre du signal échantillonné est le suivant :

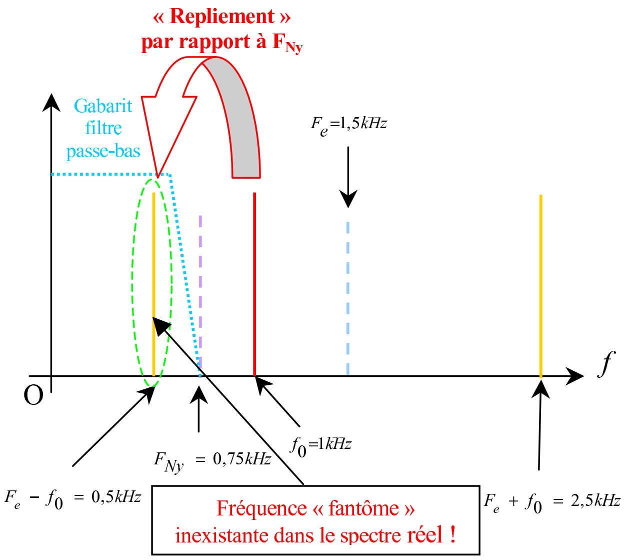

FIGURE II.9 – Phénomène de repliement de spectre pour un signal sinusoïdal

Il présente une raie dans la zone filtrée $\left[0, F_{Ny} = \frac{F_e}{2} = 0{,}75kHz\right]$ à la fréquence $F_e - f_0 = 1{,}5kHz - 1kHz = 0{,}5kHz$. Cette raie n'existe évidemment pas dans le signal originel, et est symétrique de la raie du signal à f_0 par rapport à la fréquence de Nyquist ; en effet, la fréquence médiane entre cette raie fantôme et la fréquence f_0 est :

$$\frac{(F_e - f_0) + f_0}{2} = \frac{F_e}{2} = F_{Ny}$$

A retenir : Si $F_e < 2f_0$ alors il apparait $f_{\text{fantôme}}$ symétrique de f_0 par rapport à $F_{Ny} = \dfrac{F_e}{2}$:

On parle de **repliement de spectre** par rapport à la fréquence de Nyquist.

NB : dans le cas d'un signal non périodique (spectre continu), la situation de repliement correspond à l'allure spectrale suivante :

Remarque I-2: FILTRAGE ANTI-REPLIEMENT

Pour les signaux périodiques dont le spectre est non borné supérieurement, on procède généralement à un filtrage passe-bas avant échantillonnage appelé FILTRAGE ANTI-REPLIEMENT. Ceci permet de poser *arbitrairement* une borne supérieure au spectre, i.e. de définir F_{Max} afin d'assurer le respect rigoureux de la condition de Nyquist-Shannon lors du choix de la fréquence d'échantillonnage.

II Notions de base sur le filtrage numérique des signaux

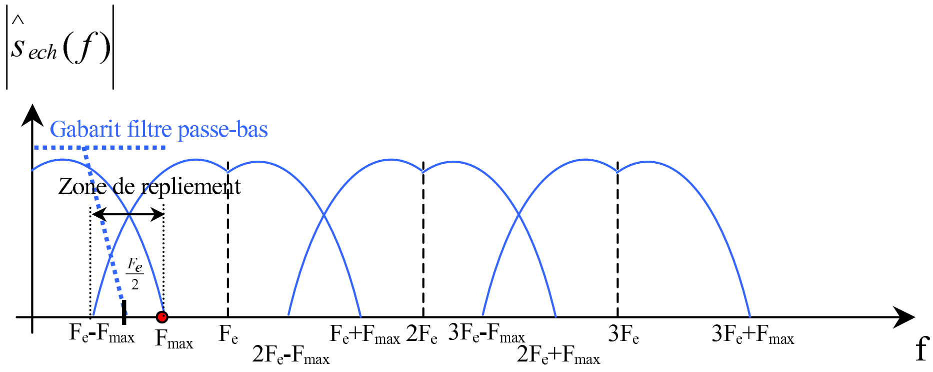

FIGURE II.10 – Phénomène de repliement de spectre

II.1 Principe

La majorité des dispositifs modernes de traitement et analyse de signal, notamment en Hifi (élimination du bruit, enrichissement sélectif du spectre graves-médiums-aigües, effets de réverbération etc...) sont des dispositifs numériques travaillant donc sur des signaux préalablement échantillonnés. Ceci permet une latitude totale dans la nature des traitements que l'on peut faire subir aux signaux puisqu'ils sont réalisés par des algorithmes programmés sur ordinateur.

Deux opérations de base du traitement numérique du signal en Hifi sont les filtrages passe-haut (élimination du bruit très basse fréquence type "Rumble"), et passe-bas (élimination de sifflements haute fréquence).

Question : quel est le principe de filtrage d'un signal échantillonné ?

2 méthodes : [filtrage temporel
filtrage spectral/fréquentiel

La suite expose les méthodes de filtrage numériques temporel et fréquentiel dont les principes sont résumés dans les synopsis ci-dessous :

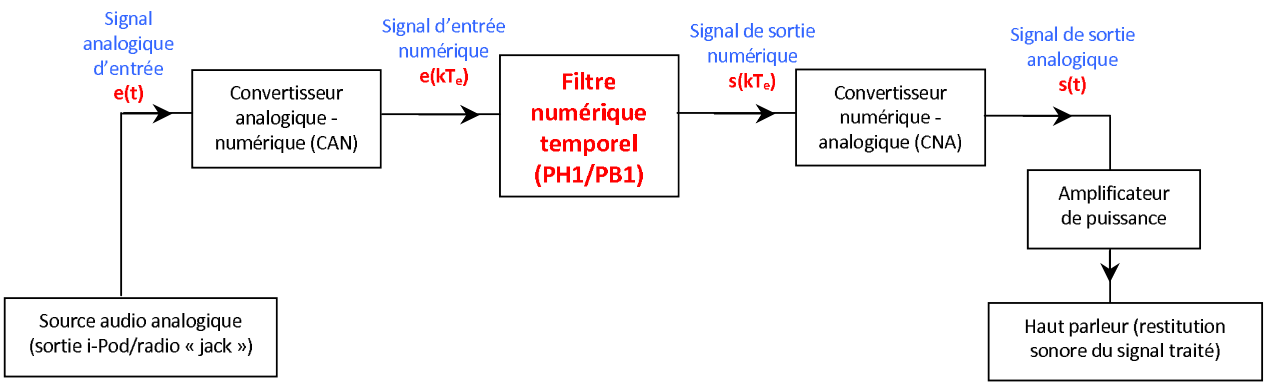

FIGURE II.11 – Chaine complète de filtrage numérique temporel d'un signal audio

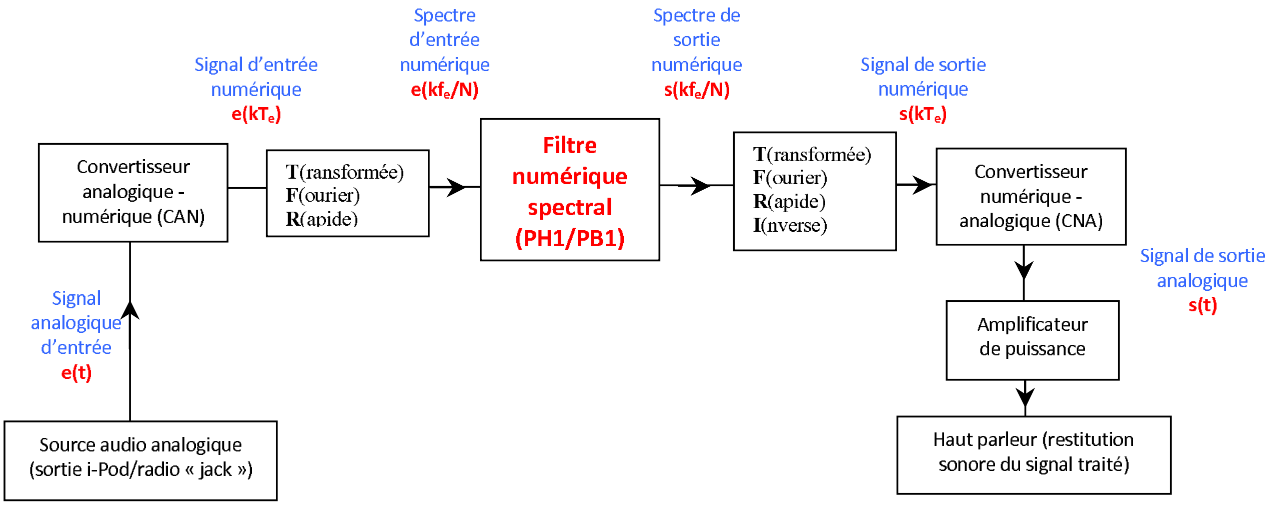

FIGURE II.12 – Chaine complète de filtrage numérique frequentiel d'un signal audio

II.2 Mise en oeuvre

a - Filtrage numérique temporel (Expérience de cours/Simulation Python)

Le signal à traiter présente deux composantes sinusoidales l'une BF à $f_b = 100\ Hz$ et l'autre HF à $f_h = 4\ kHz$:

$$e(t) = U_{0_b}\cos(2\pi f_b t) + U_{0_h}\cos(2\pi f_h t)$$

Objectif : **On se propose de réaliser par exemple l'élimination de la composante HF à l'aide d'un filtrage numérique temporel passe-bas.** Les différentes étapes que vous exécuterez en TP sont les suivantes :

- Acquisition du signal par conversion analogique-numérique (CAN) avec la fréquence d'échantillonnage $f_e = \dfrac{1}{T_e} = \geq 100\ kHz > 2f_h$ (cf th. de Shannon) avec LatisPro $\longrightarrow e(kT_e) \quad (k \in \mathbb{N})$.
- Enregistrement du signal échantillonné au format .csv (séparation des données par une virgule), sous forme de couples $(kT_e, e(kT_e))$
- Lecture du fichier .csv et **Filtrage numérique temporel** par un programme Python du signal échantillonné $e(kT_e)$, par exemple **passe-bas 1er ordre** $e(kT_e) \overset{\text{passe bas}}{\Longrightarrow} s(kT_e)$
- Enregistrement par python du signal filtré $s(kT_e)$ au format .csv
- Ouverture du fichier signal filtré par LatisPro
- Conversion numérique-analogique (CNA) du signal $s(kT_e) \longrightarrow s(t)$ et restitution par HP.

■ CALCUL DE LA "FORMULE" DE FILTRAGE DU PASSE-BAS $e(kT_e) \overset{\text{PASSE BAS}}{\Longrightarrow} s(kT_e)$

$$s(k+1) = +s(k) + \omega_c \underbrace{\int_{kT_e}^{(k+1)T_e} [e(t) - s(t)] \cdot dt}_{\text{aire sous la courbe} \Rightarrow \text{méthode des trapèzes}} \tag{II.1}$$

La fonction de transfert d'un passe bas 1[er] ordre passif de pulsation de coupure ω_c est de forme :

$$H(j\omega) = \frac{1}{1 + j\dfrac{\omega}{\omega_c}}$$

dont l'équation différentielle correspondante est :

$$\frac{ds(t)}{dt} + \omega_c \cdot s(t) = \omega_c \cdot e(t)$$

On intègre cette équation différentielle entre les instants "discrets" kT_e et $(k+1)T_e$ et l'on réordonne les deux membres pour obtenir :

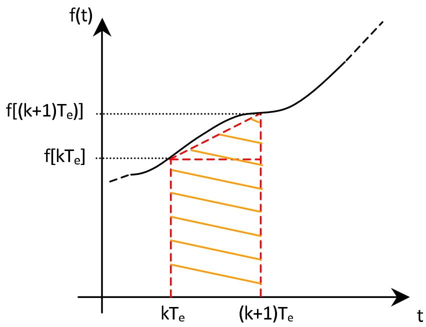

FIGURE II.13 – Intégration par méthode des trapèzes

or l'intégrale entre deux instants kT_e et $(k+1)T_e$ d'une fonction $f(t)$ peut être approchée par méthode des trapèzes avec :

$$\int_{kT_e}^{(k+1)T_e} f(t)\cdot dt \simeq \overbrace{T_e \cdot f(kT_e)}^{\text{aire du rectangle inférieur}} + \overbrace{\frac{T_e}{2}\left[f[(k+1)T_e] - f(kT_e)\right]}^{\text{aire du triangle supérieur}} = \frac{T_e}{2}\left[f[(k+1)T_e] + f(kT_e)\right]$$

Compte tenu de ceci, l'équation II.1 devient :

$$s(k+1) = s(k) + \frac{\omega_c T_e}{2}\left[e(k) + e(k+1) - s(k) - s(k+1)\right]$$

soit finalement : $$\boxed{s(k+1) = \frac{2-\omega_c T_e}{2+\omega_c T_e}\cdot s(k) + \frac{\omega_c T_e}{2+\omega_c T_e}\left[e(k) + e(k+1)\right]}$$

Cette dernière relation de récurrence permet d'obtenir le signal numérique (échantillonné) de sortie du filtre.

Le code Python complet sera commenté "en live" :

Script Python Filtrage numérique passe-bas et passe haut 1er ordre

tiny

```
# -*- coding: utf-8 -*-
from math import *
import numpy as np
import matplotlib.pyplot as plt
#ouverture du fichier de donnees source
mesdonnees=open('Donnees_formatees_CSV.csv','r')
#Donnees numeriques
Te=0.00001 #PÃ©riode d'Ã©chantillonnage
wc=2*np.pi*200.0 #Pulsation de coupure des passe-bas et passe-haut
#Tableaux vierges pour signal entree pour enregistrement de 4096 donnees
t=np.zeros(4096)
e=np.zeros(4096)
#Lecture en-tete du fichier de donnees
entete=mesdonnees.readline().rstrip('\n\r').split(',')

#Initialisation d'un compteur et du max d'Ã©chelle graphique
k,max=(0,0)
#Boucle de construction de e par lecture du fichier source
for Ligne in mesdonnees:
    tL,eL=Ligne.rstrip('\n\r').split(",") #Lit la ligne, supprime espace
#et retour chariot, coupe a la virgule et stocke le temps et la valeur signal
    t[k]=float(tL) # affectation de la kiÃ¨me valeur du temps
    e[k]=float(eL) # affectation de la kiÃ¨me valeur du signal
    if e[k]>max:
        max=1.05*e[k] #Max est une variable d'ajustement automatique de l'echelle
    k+=1
#Fermeture du fichier source
mesdonnees.close()
#Tableaux vierges pour signal sortie pour enregistrement 4096 donnees
spb1=np.zeros(4096)
sph1=np.zeros(4096)
#ouverture des fichiers de donnees sortie
masortie_PB1=open('Donnee_sortie_PB1_CSV.csv','w')
masortie_PH1=open('Donnee_sortie_PH1_CSV.csv','w')
#Ecriture des signaux de sortie des filtres par rÃ©currence
for k in range(4095):
    spb1[k+1]=((2-wc*Te)/(2+wc*Te))*spb1[k]+((wc*Te)/(2+wc*Te))*(e[k]+e[k+1])#PB
    sph1[k+1]=((2-wc*Te)/(2+wc*Te))*sph1[k]+(2/(2+wc*Te))*(e[k+1]-e[k])#PH
    masortie_PB1.write(str(t[k])+','+str(spb1[k])+'\n') #Ecriture de ligne dans le fichier de sortie Passe
     bas1 aprÃ¨s conversion en chaine de caractere
    masortie_PH1.write(str(t[k])+','+str(sph1[k])+'\n') #Ecriture de ligne dans le fichier de sortie passe
     haut1 aprÃ¨s conversion en chaine de caractere
#Fermeture des fichiers de sortie passe-bas1 et passe-haut1
masortie_PB1.close()
masortie_PH1.close()
#Trace evolution des tensions entree et sortie
plt.grid()
plt.xlabel(r'$t(s)$', fontsize=10)
plt.ylabel(r'$s(t)\␣(en\␣V)$', fontsize=10, rotation=0)
plt.title(r"Tension␣de␣sortie␣du␣filtre",size=20)
plt.axis([t[2048],t[4095],-max,max])
#entree=plt.plot(t,e)
sortie_PB1=plt.plot(t,spb1)
#sortie_PH1=plt.plot(t,sph1)
plt.show()
```

Listing II.1 – Sources_Python/Filtrage_numerique_version_2_boucles_pour_compil_latex.py

NB : le fichier Python est disponible sur le site MP3, et libre de modification/amélioration si le cœur vous en dit.

■ CALCUL DE LA "FORMULE" DE FILTRAGE DU PASSE-HAUT

A faire en live à titre d'exercice.

b - Filtrage numérique spectral

Même exemple mais cette fois **par la technique fréquentielle** $\Longrightarrow$ cf réalisation en TP à l'aide d'un programme python fourni et **document capacité numérique.**

PRINCIPE :

- On réalise la TFR du signal d'entrée ⇒ on dispose des amplitudes complexes $\underline{d_{n_e}}$ des composantes de ce signal.
- On calcule à l'aide de la fonction de transfert et des $\underline{d_{n_e}}$ les amplitudes complexes $\underline{d_{n_s}}$ des composantes du signal de sortie.
- On reconstitue le signal par transformée de Fourier inverse.

$$\underline{s}\left(k\frac{F_e}{N}\right) = H\left(j2\pi k\frac{F_e}{N}\right)\cdot \underline{e}\left(k\frac{F_e}{N}\right) \quad \Longrightarrow \quad \left[\begin{array}{l} \left|s\left(k\frac{F_e}{N}\right)\right| = G\left(k\frac{F_e}{N}\right) \times \left|e\left(k\frac{F_e}{N}\right)\right| \\ \\ \varphi_s\left(k\frac{F_e}{N}\right) = \varphi_e\left(k\frac{F_e}{N}\right) + arg\left[H\left(j2\pi k\frac{F_e}{N}\right)\right] \end{array}\right.$$

avec N nombre total d'échantillons prélevés et $k \in [0, \lfloor\frac{N}{2}\rfloor]$ (afin d'assurer $f \in \left[0, F_{Ny} = \frac{F_e}{2}\right]$).