LP 23 - Traitement du Signal. Etude Spectrale

5 Décembre 2017

le spectre d'Alexandre & Pierre Comelli

Traitement de la phasmophobie (peur des fantômes) - Honoraires : 60 euros/séance
A.Mo-PAS DE PUB !!

Niveau : L3

Commentaires du jury

Fourier est central à cette leçon [2010] mais on ne peut réduire la leçon à cela [2016]. Cette leçon ne peut se réduire au filtrage [2017] ou à un catalogue de système de traitements du signal [2015, 2008]. On peut étudier les systèmes non-linéaires [2010] et numériques [2015]. Le titre de cette leçon ayant beaucoup varié, les commentaires du jury aussi.

Bibliographie

- ✍ *L'essentiel en théorie et traitement du signal*, **Yvan Duroc** ⟶ Complet et très synthétique sur Traitement du Signal et l'étude Spectrale. Parfait si vous voulez juste des rappels.
- ✍ *Traitement des signaux et acquisition de données*, **Francis Cottet** ⟶ Complet et détaillé *ma non troppo* sur Traitement du Signal et Etude Spectrale. Très bien si le Duroc est trop synthétique pour vous.
- ✍ *Physique tout-en-un PSI-PSI** **Sanz** ⟶ Détails pour la transformation de Fourier et les séries de Fourier
- ✍ *BUP 872 - Cryptage du son et traitement numérique*, **IDDA** ⟶ Toute la description du chiffrement/déchiffrement du signal sonore de Canal+. Ordres de grandeur. Rappels sur le traitement du signal. Explications sur l'utilisation des logiciels pour obtenir les fichiers sons. Cette leçon se base quasi totalement sur ce BUP.
- ✍ *Expériences d'électronique*, **Duffait** ⟶ Pour le filtre RC

Prérequis

- ➢ Propriétés de la Transformée de Fourier
- ➢ Distribution de Dirac
- ➢ Bases de l'électronique

Expériences

- Filtre RC
- construction de signaux via python

Table des matières

Introduction

Je vais vous parler d'un temps que les plus de vingt ans connaissent. Vous vous rappelez de la chaîne Canal+ qui émettait en crypté (*Nota Bene* : il ne faut pas dire signal crypté, c'est un abus de langage, mais je le ferais quand même dans ce poly). L'image était brouillée et le son donnait ceci :

Exemple d'un son brouillé
Matériel : ordinateur, hauts-parleurs, fichier son chiffré 1, VLC
Lire le fichier audio, montrant en évidence le caractère incompréhensible du signal chiffré envoyé à l'époque par Canal+ si l'on n'avait pas de décodeur.
Si vous n'avez pas ce fichier son, créez-en un à l'aide de l'annexe.

On va donc s'intéresser à comment transformer ce son chiffré pour le rendre compréhensible.

Pour cela, on va d'abord cherche à comprendre où est l'information dans le signal, ce via l'étude spectrale.

1 Etude Spectrale

1.1 Signal pur

✍ *Physique Tout-en-un PSI-PSI* p.827*, **Sanz**

Voyons déjà ce que donne un signal pur, avec l'exemple d'un son pur de diapason.

Son pur avec un diapason
Matériel : Oscilloscope, diapason, marteau à diapason, micro, ampli de micro
Mettre le micro dans la cavité de résonance du diapason. Le relier via un ampli à l'oscilloscope.
Frapper le diapason avec le marteau. Visualiser en fonction du temps une sinusoïde parfaite sur l'oscilloscope. Vérifier que la fréquence du signal obtenu corresponde bien à celle tabulée du diapason.

Dans une représentation dans le temps, on voit que ce signal pur est une sinusoïde fréquence constante :

$$x(t) = A * sin(2\pi * f_0 * t) \tag{1}$$

Ce signal pur, on peut le représenter en fréquence grâce à la transformée de Fourier, définie comme :

$$X(f) = TF(x(t))(f) = \int_{-\infty}^{\infty} x(t) e^{-2i\pi ft} \, \mathrm{d}t \tag{2}$$

Cette TF est définie ainsi si elle existe, ce qui est le cas pour les signaux physiques vu qu'ils sont forcément d'énergie finie. Sinon, on peut raisonner avec les puissances des signaux ou avec les distributions.
Pour ce signal pur, cela donne

$$X(f) = \frac{A}{2i}(\delta(f - f_0) - \delta(f + f_0)) \tag{3}$$

Avec δ la distribution de Dirac.
On a alors l'énergie du signal

$$E = \int_{-\infty}^{\infty} |x(t)|^2 \, \mathrm{d}t \tag{4}$$

Le théorème de Parseval nous donne

$$E = \int_{-\infty}^{\infty} |X(f)|^2 \, \mathrm{d}f \tag{5}$$

On a alors la densité spectrale d'énergie [dse] du signal $S_x(f) = |TF(x)(f)|^2$. Avec par exemple celle du signal pur ici employé :

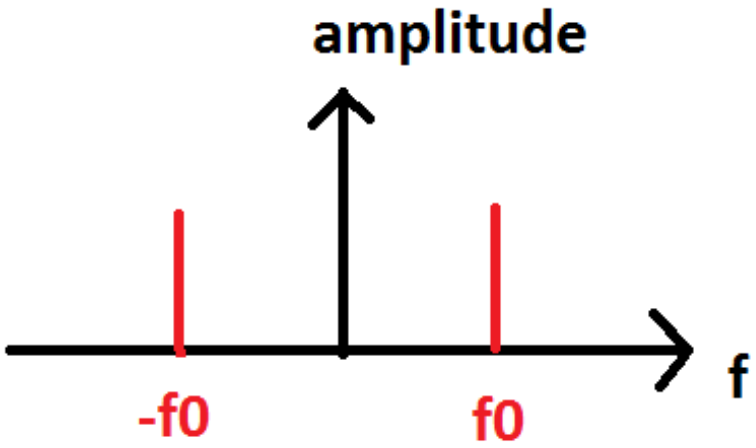

FIGURE 1 – Spectre d'un signal sinusoïdal de fréquence f0

A priori, il n'y a pas de sens à représenter la densité spectrale d'énergie pour des fréquences négatives, mais cela est utile dans certains cas lors des phases de traitement du signal pour éviter les erreurs.
On peut également raisonner en puissance avec

$$P = \lim_{T\to+\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} |x(t)|^2 \, \mathrm{d}t \tag{6}$$

et on aura alors la densité spectrale de puissance [dsp] $S_{px}(f) = \lim_{T\to+\infty} \frac{1}{T}|TF(x)(f)|^2$.

Attention, quand on parle de spectre du signal, cela peut correspondre à la densité spectrale d'énergie ou de puissance, ou le module simple de la TF. Le terme spectre peut même correspondre au spectre complexe.

Mais dans la vie, on n'a pas que des signaux purs. On ne s'exprime pas par exemple qu'en diapasons. Il existe aussi des signaux complexes.

1.2 Signal complexe périodique

Analysons ces signaux complexes. S'ils sont de période $T_s = \frac{1}{f_s}$, on peut les décomposer en une somme de sinusoïdes.

Construction d'un signal complexe
Matériel : ordinateur, programme python (cf annexe)
Avec le programme, créer un signal complexe en tant que somme de sinusoïdes.
Il est aussi possible de représenter le spectre associé au signal complexe pour mettre en jeu le côté spectre complexe = somme de spectres simples.

On peut mathématiquement décomposer ce signal complexe périodique en série de Fourier :

$$x(t) = \frac{A_0}{2} + \sum_{n=1}^{\infty} (a_n * cos(2n\pi f_s t) + b_n * sin(2n\pi f_s t)) \tag{7}$$

$$x(t) = C_0 + \sum_{n=1}^{\infty} (C_n * cos(2n\pi f_s t + \phi_n) \tag{8}$$

avec les coefficients de Fourier :

$$a_n = \frac{2}{T} \int_{T}^{T+T_s} x(t) cos(2n\pi f_s t) \, \mathrm{d}t, n \in N \tag{9}$$

$$b_n = \frac{2}{T}\int_{T}^{T+T_s} x(t)sin(2n\pi f_s t)\,\mathrm{d}t, n \in N^* \tag{10}$$

$$a_0 = \frac{2}{T}\int_{T}^{T+T_s} x(t)\,\mathrm{d}t \tag{11}$$

$$C_0 = \frac{a_0}{2} \tag{12}$$

$$C_n = \sqrt{a_n^2 + b_n^2}; cos(\phi_n) = \frac{a_n}{C_n}; sin(\phi_n) = \frac{b_n}{C_n}; n \in N* \tag{13}$$

Avec C_0 la valeur moyenne de x(t), la composante à n=1 la composante fondamentale du signal et les composantes à n>1 les harmoniques.
Par linéarité de la TF, on alors le spectre d'un signal complexe comme la somme des spectres des signaux composant le signal complexe. Ce spectre donne l'intensité de chaque composante fréquentielle du signal.

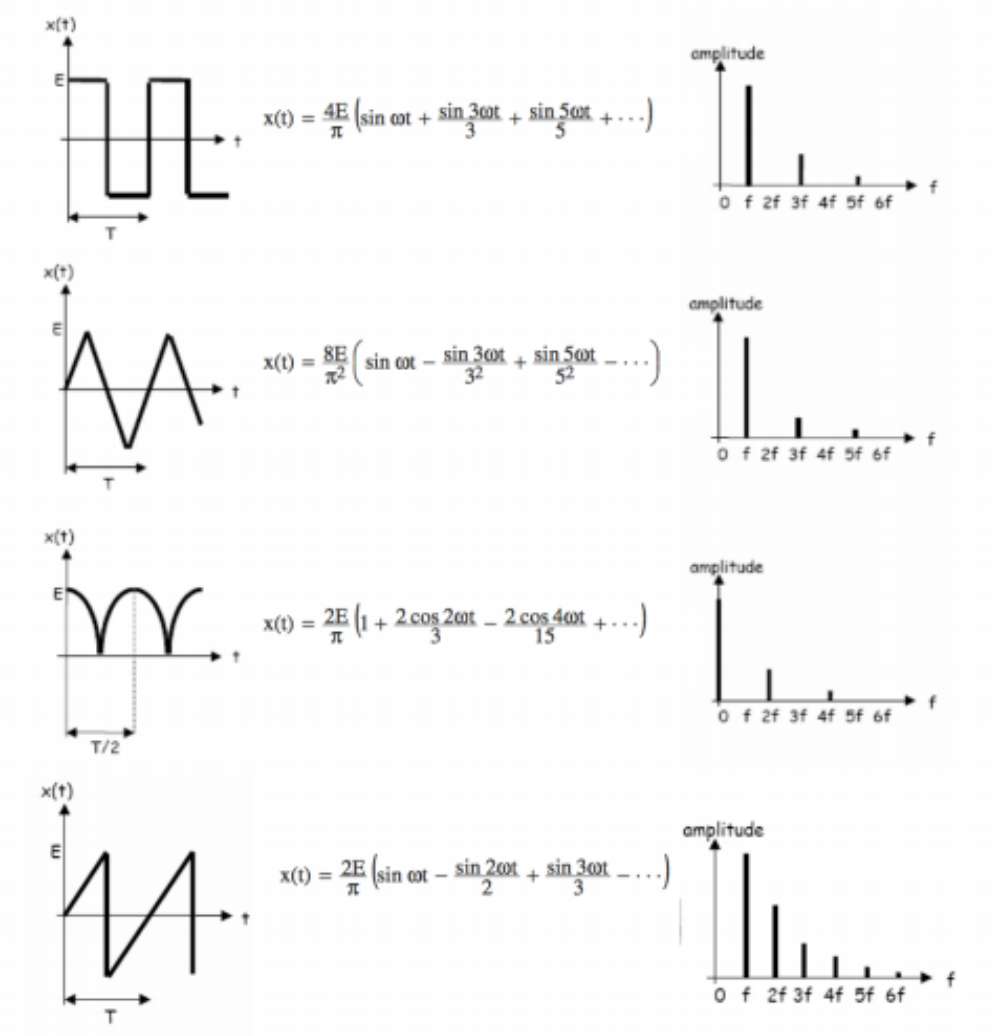

FIGURE 2 – Différents signaux et leurs spectres associés

Maintenant que l'on a la représentation en fréquence d'un signal, on va pouvoir le manipuler

Analysons la sensibilité de l'oreille humaine. Celle-ci entend les sons compris entre 20 et 20kHz, avec une bonne sensibilité de 100 à 10kHz et un pic de sensibilité à 1kHz.

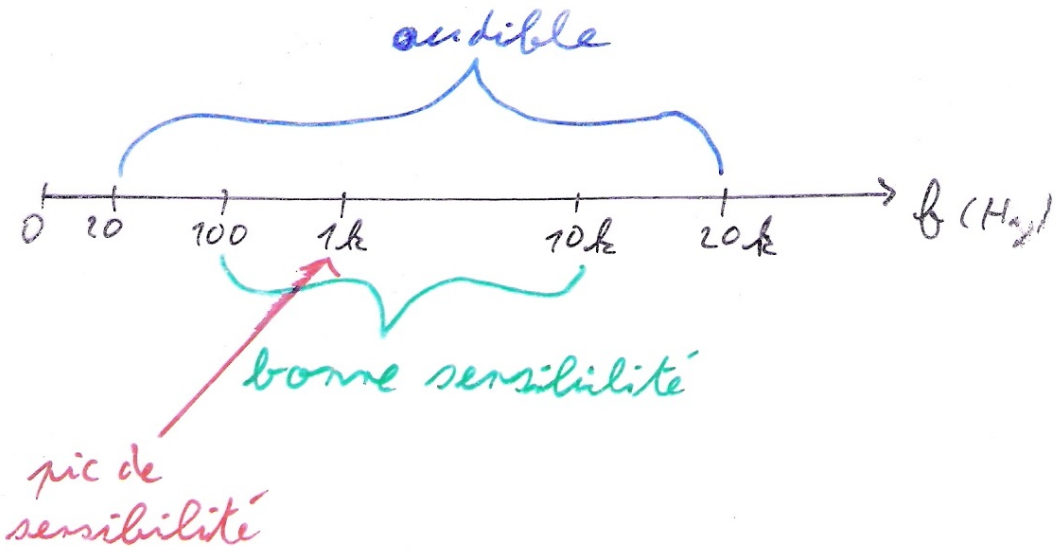

FIGURE 3 – Plage d'audition en fréquence de l'oreille humaine (modèle ordianire)

Observons la dsp de la sublime chanson qu'est la Canzone de Marinella de Fabrizio dé André :

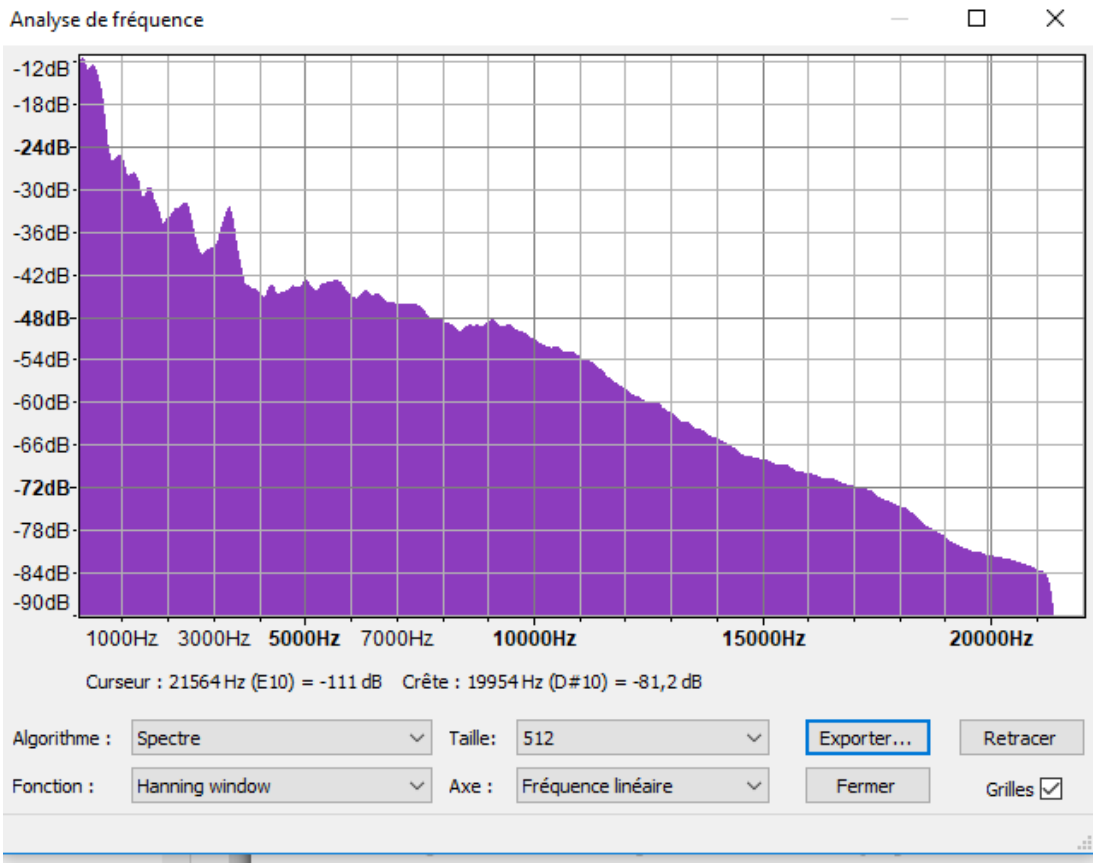

FIGURE 4 – Spectre d'une chanson extraordinaire

On peut remarquer que l'on a un maximum d'intensité proche de 1kHz, puis que la dsp diminue progressivement pour être presque nulle après 20kHz.

Observons maintenant la dsp du son Canal+ chiffré :

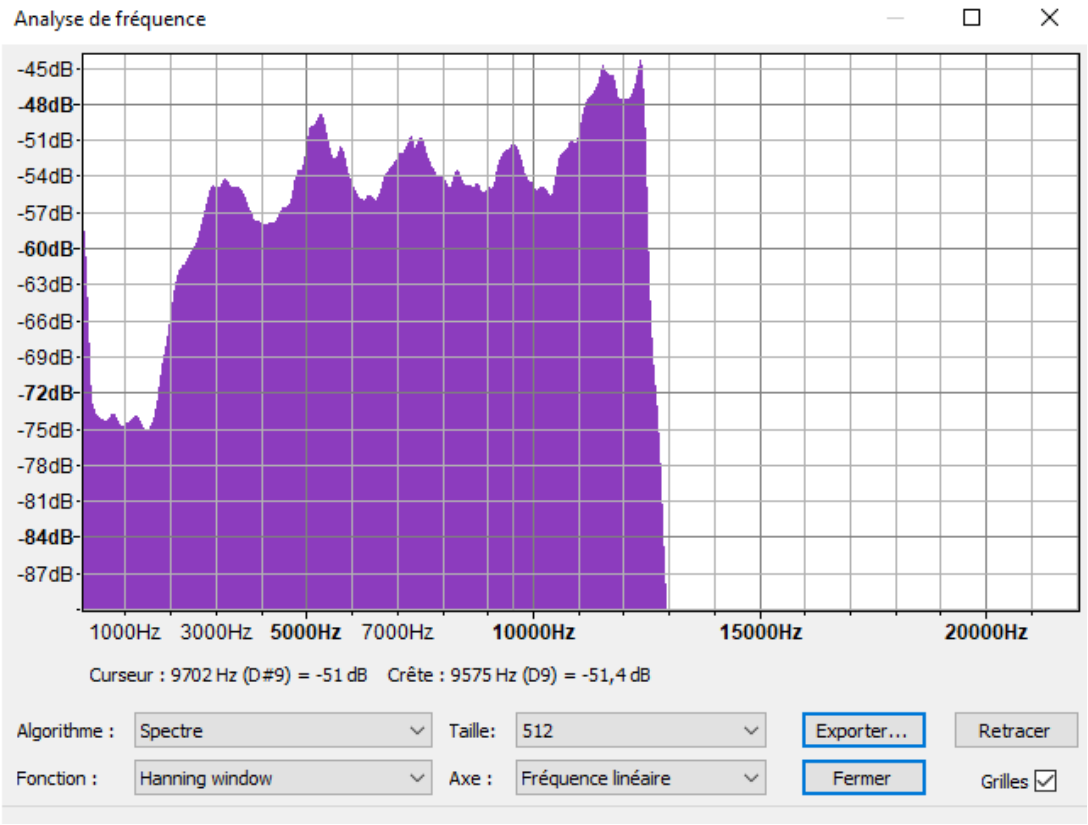

FIGURE 5 – Spectre du son Canal+ chiffré

On peut remarquer que la dsp est très faible autour de 1kHz, puis qu'elle augmente progressivement avant de s'arrêter brusquement à 12800Hz.

En fait, pour déchiffrer le son, on doit procéder à une modulation en amplitude du signal

2 Modulation d'amplitude

D'habitude, la modulation d'amplitude se fait pour faciliter la transmission de signaux. Ici, elle sert à déchiffrer un signal.

La porteuse $p(t) = B * cos(2 * pi * f_p * t)$ est un signal sinusoidal. Soit x(t) le signal informatif. Un signal modulé en amplitude y(t) est un signal pour lequel l'amplitude de la porteuse varie en fonction de x(t). On réalise cela en multipliant les deux signaux :

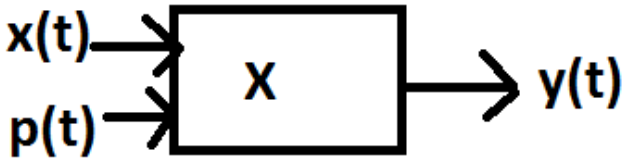

FIGURE 6 – Représentation d'une modulation en amplitude

Ex avec un signal pur : $x(t) = A * cos(2 * pi * f_x * t)$. Alors :

$$y(t) = A * B * cos(2 * pi * f_p * t) * cos(2 * pi * f_x * t) = \frac{A * B}{2} * (cos(2 * pi * t * (f_p - f_x)) + cos(2 * pi * t * (f_p + f_x)) \quad (14)$$

dont la représentation spectrale est :

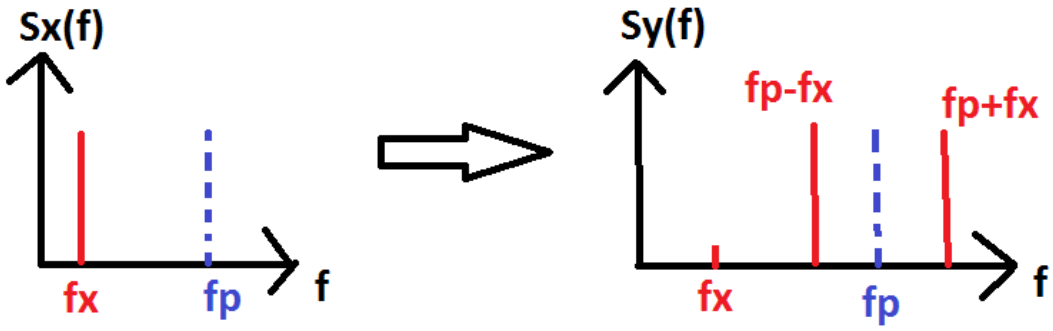

FIGURE 7 – Spectre d'un signal sinusoïdal de fréquence fx - Spectre de ce signal modulé par une porteuse fp

Donc, pour notre son chiffré, on a $f_p = 12800Hz$ (voir annexe sur le pourquoi de cette valeur), on obtient le signal de densité spectrale :

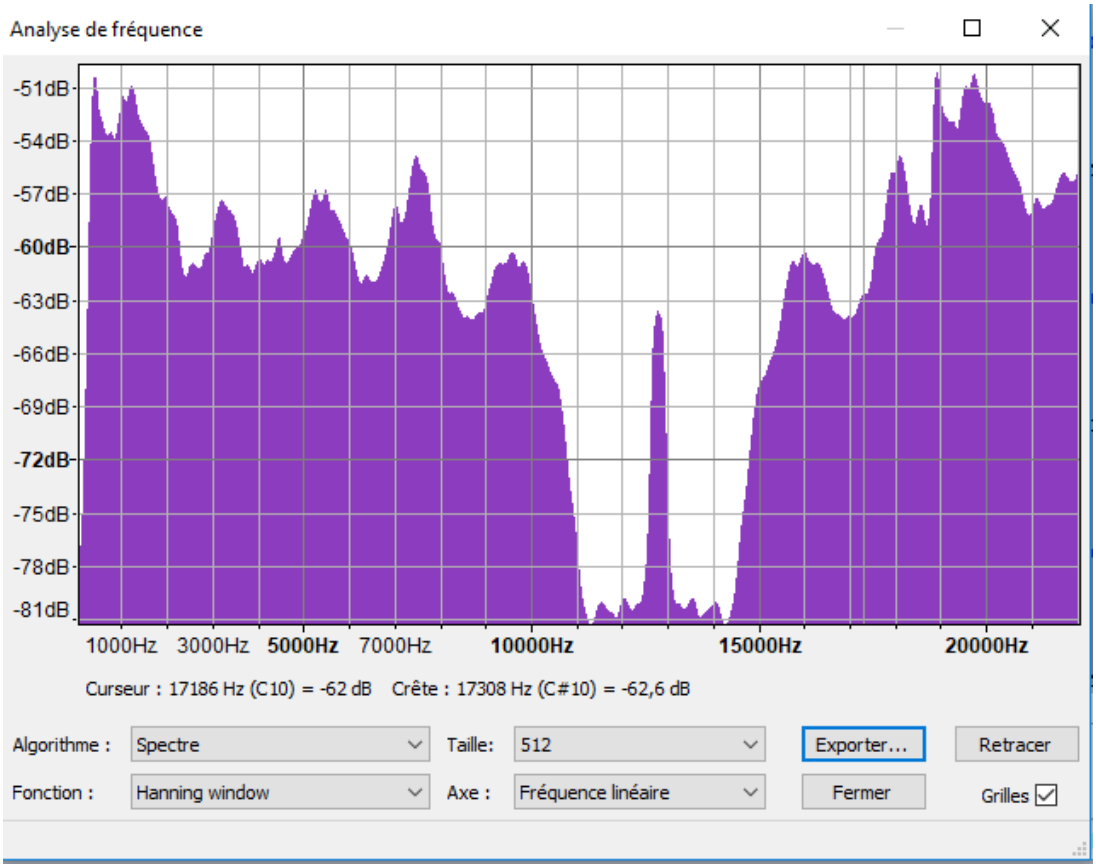

FIGURE 8 – Spectre du son Canal+ après modulation

On fait alors écouter le son déchiffré et on obtient un signal compréhensible.

> *Mais on peut voir qu'il y a un souci : le spectre somme, même s'il est dans les fréquences peu audibles, déforme le signal. Il va donc falloir l'éliminer via un filtrage.*

3 Filtrage

Filtrage RC

✍ Duffait p. 141 ⏲

Matériel :

plaquette

oscilloscope

GBF

condensateur C=10nF

résistance R=12,4kΩ

Observer à l'oscilloscope le signal fourni par le GBF et celui aux bornes du condensateur.

Partir d'un signal basse fréquence en entrée. Observer que la sortie reste identique à l'entrée.

Augmenter progressivement la fréquence en entrée. Observer que la sortie reste d'abord à peu près identique à l'entrée puis qu'elle diminue en amplitude et se déphase. Au delà de 12800Hz, constater que la sortie est quasiment nulle et déphasée de 90.

Une autre possibilité est de tracer directement le diagramme de Bode du filtre

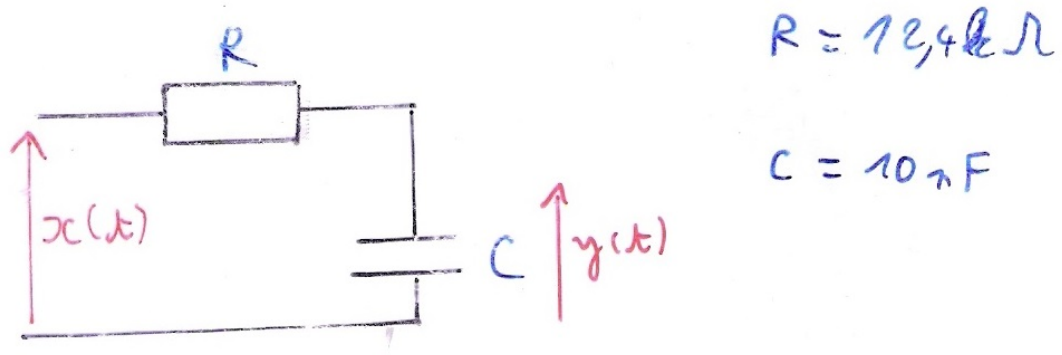

FIGURE 9 – Schéma électrique du cricuit RC - filtre passe-bas

On a ici un Système Linéaire Invariant dans le Temps (SLIT) qui agit par pondération des coefficients de Fourier du signal (et donc de son spectre). En fait, on a :

$$y(t) = (h \otimes x)(t) \tag{15}$$

Soit, dans le domaine fréquentiel/espace de Fourier :

$$Y(f) = TF(h \otimes x)(f) \tag{16}$$

On utilise le théorême de Plancherel / la propriété de la convolution de la TF et on obtient :

$$Y(f) = TF(h)(f) * TF(x)(f) = H(f) * X(f) \tag{17}$$

Aparté — Théorême de Plancherel

$$z(t) = (x \otimes y)(t) = \int_R x(\tau)y(t-\tau)\,\mathrm{d}\tau \tag{18}$$

$$Z(f) = \int_R (\int_R x(\tau)y(t-\tau)\,\mathrm{d}\tau)e^{-2i\pi ft}\,\mathrm{d}t = \int_R x(\tau) * e^{-2i\pi f\tau} \int_R y(t-\tau)e^{-2i\pi f(t-\tau)}\,\mathrm{d}\tau\mathrm{d}t \tag{19}$$

On pose $t' = t - \tau$ et on a :

$$Z(f) = \int_R x(\tau)e^{-2i\pi f\tau}\,\mathrm{d}\tau * \int_R y(t')e^{-2i\pi ft'}\,\mathrm{d}t' = X(f) * Y(f) \tag{20}$$

fin aparté

On a la fonction de transfert du SLIT :

$$H(f) = \frac{Y(f)}{X(f)} \tag{21}$$

Dans le cadre du filtre RC, on a un filtre du premier ordre :

$$H(\omega) = \frac{1}{1 + jRC\omega}, \omega = 2\pi f \tag{22}$$

On a la pulsation de coupure :

$$\omega_c = \frac{1}{RC} \tag{23}$$

On peut construire son diagramme de Bode, avec le gain $G(f) = 20 * log(H(f))$ et le déphasage $\phi(f) = arctan(\frac{Im(H(f))}{Re(H(f))}$.
Pour cela, on détermine les comportement asymptotiques :

A basse fréquence, $\frac{\omega}{RC} << 1$, donc H(f)->1.

A haute fréquence, $\frac{\omega}{RC} >> 1$, donc $H(f) -> \frac{-j}{RC\omega}$.

On a alors (voir figure filtres).

On eut également avoir des filtres plus compliqués. Par exemple, pour une radio, on a un filtre passe-bande pour ne garder que les signaux de la fréquence de la station radio écoutée.

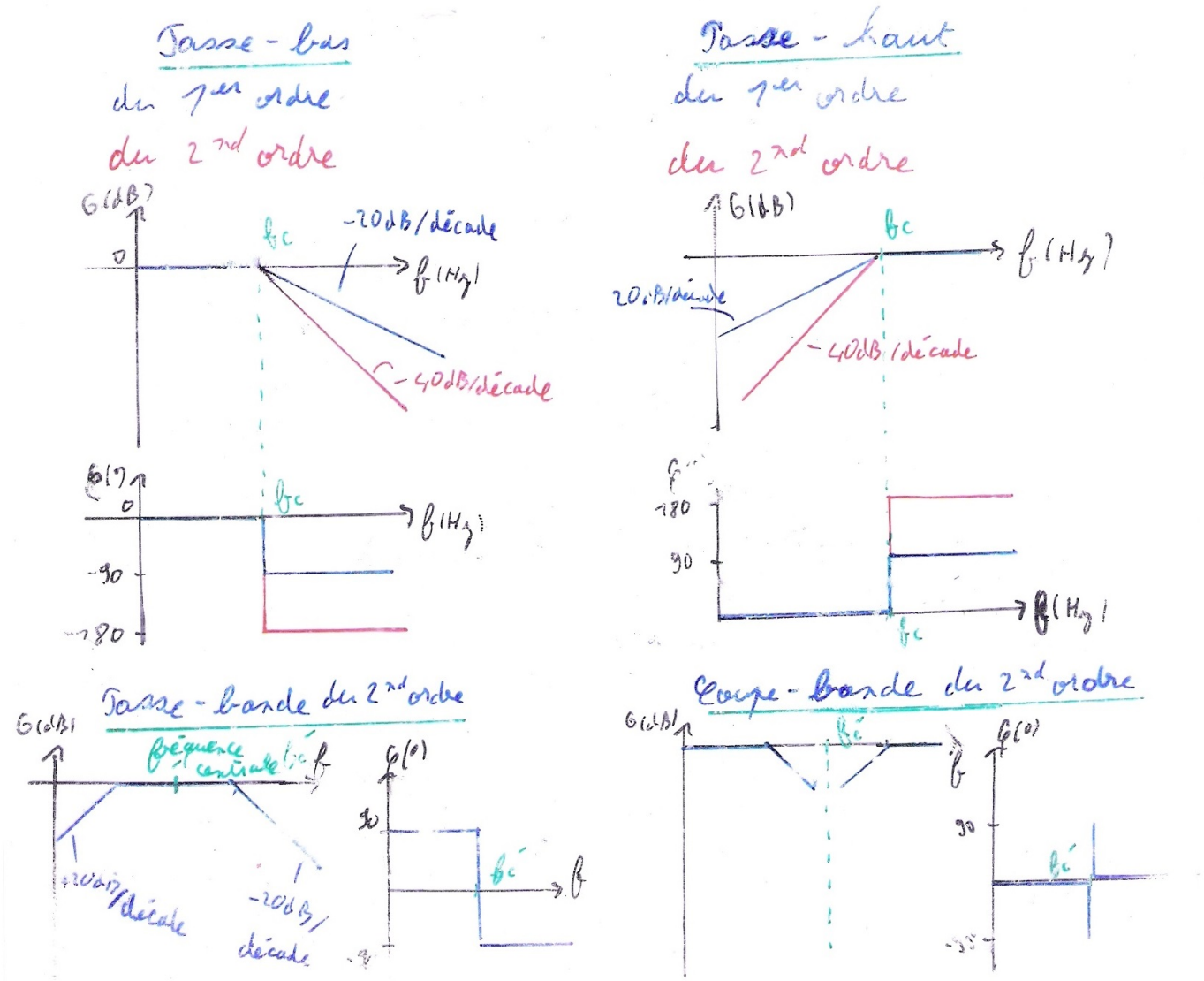

Figure 10 – Diagrammes de Bode de différents filtres

Pour le signal chiffré Canal+, comme on a procédé à une modulation du spectre à 12800Hz, et donc une recopie du signal au-delà de 12800Hz, on procède à un filtrage passe-bas de fréquence de coupure de 12800Hz. On obtient alors le spectre :

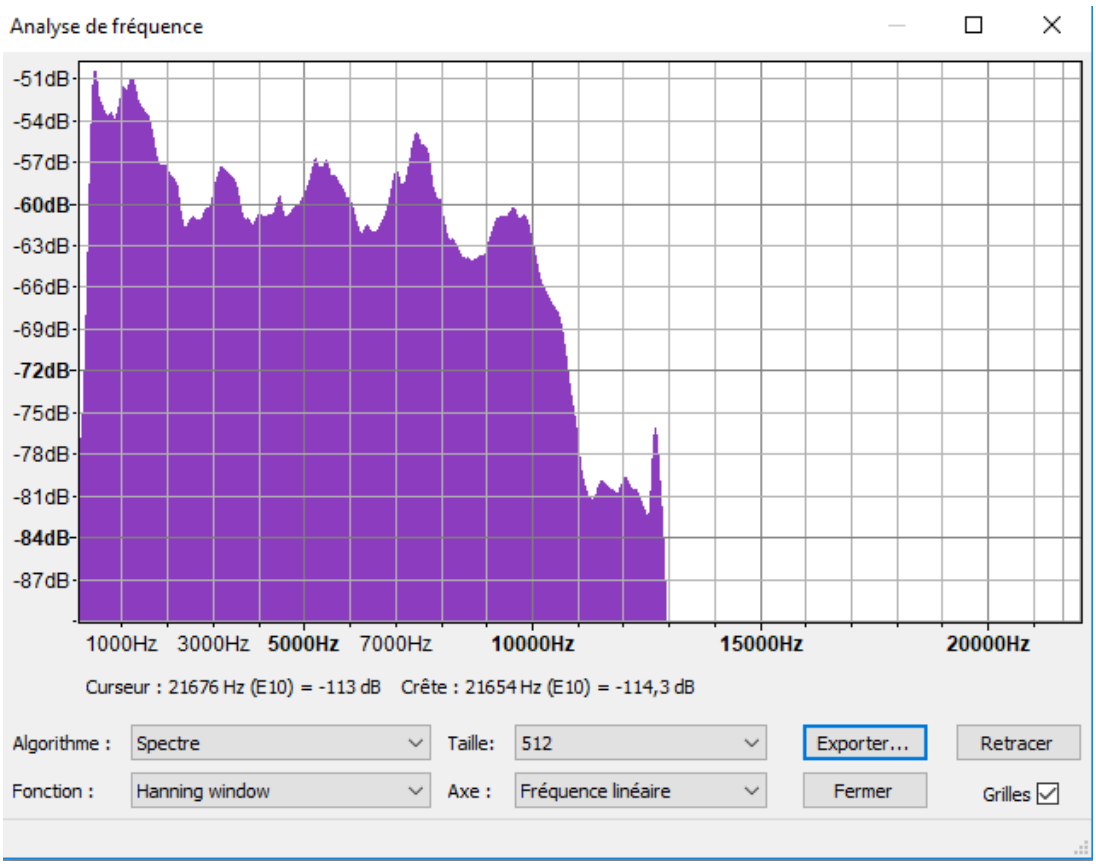

FIGURE 11 – Spectre du son Canal+ après modulation et filtrage passe-bas

On fait maintenant écouter le signal filtré. Le signal est compréhensible. On constate peu de différences avec le signal pré-filtrage car la partie filtrée était dans un domaine peu audible.

On a enfin notre signal déchiffré non déformé. Le proglème est que celui-ci est analogique. Or, supposons que l'on veuille l'écouter sur un équipement numérique. On va devoir procéder à une numérisation du signal.

4 Numérisation

4.1 Echantillonnage

Les équipements numériques ne peuvent traiter que des nombres finis et non pas des signaux continus. On va donc devoir dans un premier temps échantillonner, c'est-à-dire discrétiser dans le temps le signal continu.

Supposons que l'on échantillonne à intervalles réguliers, avec $\tau = \frac{1}{f_e}$ le pas d'échantillonnage, f_e la fréquence d'échantillonnage. On a alors le signal échantillonné x_e :

$$x_e(t) = x(t) * \sum_{n \epsilon Z} \delta(t_n \tau) = x(t) * \text{III}_\tau(t) \tag{24}$$

Avec III_τ le peigne de Dirac.

La fréquence d'échantillonnage a une influence sur l'information transmise (ou non), comme on peut le voir ici :

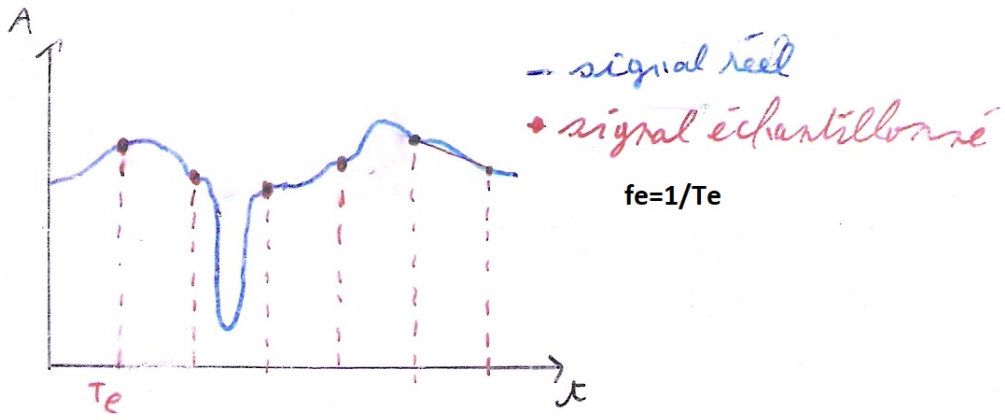

FIGURE 12 – Perte d'information provoquée par l'échantillonnage de pas Te

Pour éviter la perte d'informations, il faut respecter le critère de Shannon $f_e \geqslant 2f_max$, avec f_max la fréquence maximale du signal à échantillonner.

Mettons en évidence ce critère en étudiant le spectre du signal échantillonné :

$$X_e(f) = TF(x(t)) * \sum_{n\epsilon Z} \delta(t_n\tau))(f) \tag{25}$$

$$X_e(f) = TF(x)(f) \otimes TF(\sum_n \delta(t - n\tau))(f) \tag{26}$$

$$X_e(f) = X(f) \otimes \frac{1}{\tau}\text{III}_{\frac{1}{\tau}}(f) \tag{27}$$

$$X_e = \frac{1}{\tau}\sum_n X(f - \frac{n}{\tau}) \tag{28}$$

$$X_e = \sum_n f_e * X(f - nf_e) \tag{29}$$

Représentation spectrale :

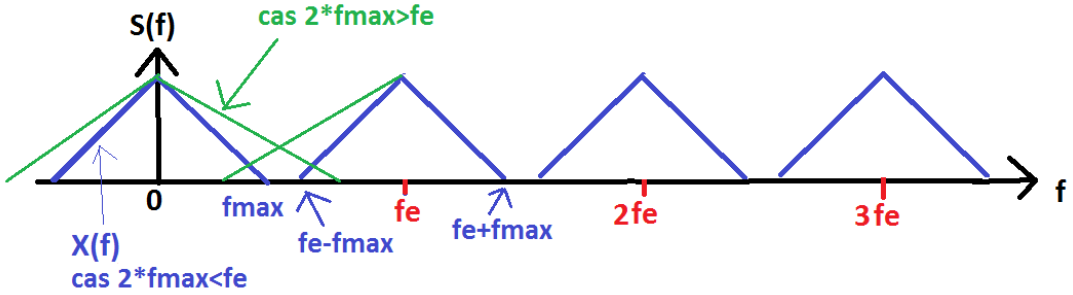

FIGURE 13 – Spectre d'un signal de fréquence maximale fmax échantillonné à fe

On peut voir que l'on repliement du spectre et perte d'information si $f_e - f_max \leqslant f_max$. D'où le critère de Shannon $f_e \geqslant 2f_max$. Il est ici indispensable de représenter les composantes positives ET négatives du spectre du signal de départ, ce pour meetre en évidence les problèmes éventuels de repliement.

Comme l'oreille humaine entend les sons jusqu'à 20kHz environ, les CD de musique étaient échantillonnés à 44100Hz pour avoir un son de bonne qualité. Pour les conversations téléphoniques, comme la qualité du son n'était pas forcément primordiale et que la contrainte était la quantité d'information (voir *infra*, l'échantillonnage se fait (faisait ?) souvent à 8kHz.

On aura donc intérêt de procéder à un filtrage passe-bas de fréquence de coupure $f_c = \frac{f_e}{2}$, avant d'échantillonner à la fréquence f_e. Cela permettra d'éviter un repliement du spectre et donc une déformation du signal.

4.2 Quantification

Mais on n'a pas tout raconté en expliquant qu'il suffisait de prendre la valeur instantanée pendant l'échantillonnage pour l'envoyer dans le système numérique.

Déjà, le système a besoin de temps pour procéder à ces opérations. Il va donc falloir procéder à la mise en mémoire de la valeur du signal, ce grâce à un bloqueur d'ordre zéro :

$$x_b = \sum_n x(n\tau) * rect_\tau(t - (n + \frac{1}{2})\tau) \tag{30}$$

Avec la fonction porte $rect_\tau(t) = 1$ si $t\epsilon[-\frac{\tau}{2}; \frac{\tau}{2}]$, 0 ailleurs.

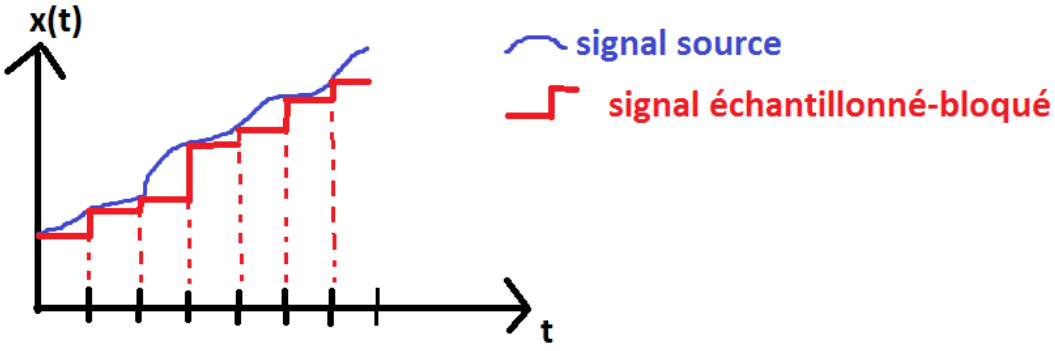

FIGURE 14 – signal échantillonné-bloqué

De plus, le numérique n'accepte qu'une certaine plage de nombre, pas de signaux continus. On va donc devoir procéder à une quantification du signal, c'est-à-dire une discrétisation en amplitude. Cette quantification sera plus ou moins précise selon le nombre de bits sur laquelle elle se fera, c'est-à-dire le nombre de valeurs possibles.

Cette discrétisation peut entraîner une perte d'information :

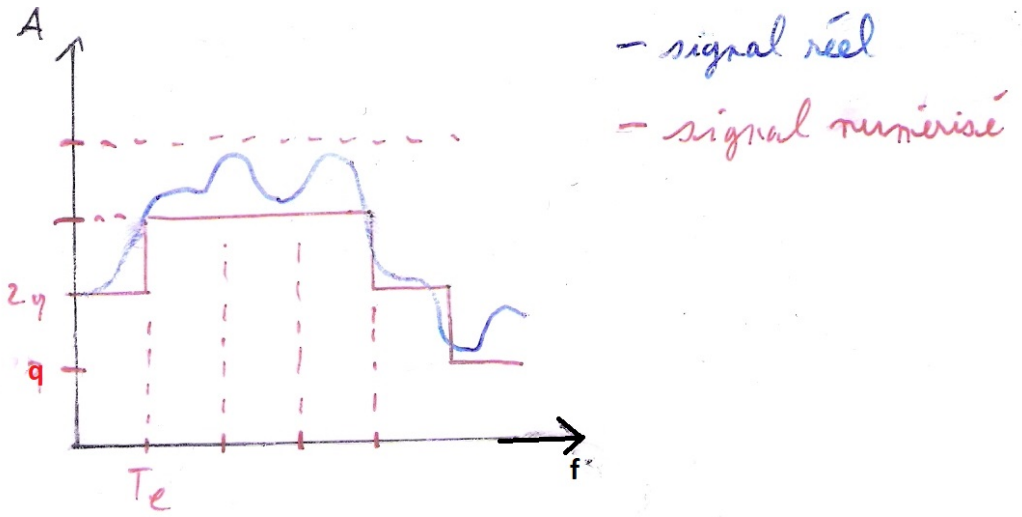

FIGURE 15 – Perte d'information provoquée par la quantification de pas q

Tout sera alors question de compromis, entre l'information que l'on veut conserver et les capacités du système.

Par exemple, pour 1 CD avec un bon son (échantillonnage à 44,1kHz, quantification sur 16 bits) :
débit d = 44100 * 16 = 705.6 kbits/s
volume en 1 minute v = débit * 60 = 42,346 10^6 bits ≈ 5,3 Mo

Pour du son MP3, numérisation à 24kHz et 8 bits
d = 192 kbits/s ; v ≈ 1,9Mo

Pour un système de surveillance généralisée des conversations téléphoniques, numérisation à 8kHz et 8 bits
d = 64 kbits/s ; v ≈ 480ko

5 Conclusion

6 Annexe

6.1 Créer ses signaux - Tracer ses spectres

✍ *BUP 872 p319*,
Canal+ a cessé d'émettre en analogique en 2011. Aujourd'hui, elle émet directement un signal numérique et les opérations de chiffrement/déchiffrement ici décrites sont caduques. Résultat, il est compliqué d'obtenir un signal chiffré Canal+ que l'on va ensuite traité.

Quelques vidéos de canal + chiffré existent encore sur internet, par exemple https ://youtu.be/cGTJEjLmx8s (attention, cet extrait ne fait pas apparaître la fréquence à 15625Hz (cf *infra*). Télécharger le son du signal avec par exemple clipconverter.cc et procéder aux opérations de déchiffrement décrites dans le BUP.

Sinon, vous pouvez vous-même créer un son Canal+ chiffré. Partez d'un fichier son normal et procédez aux mêmes étapes de déchiffrement décrites dans le BUP. Vous obtiendrez le signal chiffré (mais dépourvu de la fréquence à 15625Hz).

Et oui, le chiffrement et le déchiffrement du signal sonore de la chaîne crypté sont identiques.

Pour tracer le spectre d'un signal, vous pouvez utiliser **Goldwave** comme dans le BUP. Mais je trouve plus simple (et avec un résultat plus propre) d'utiliser le logiciel libre **Audacity** : sélectionner le fichier son concerné et aller dans Analyses -> Tracer le Spectre

6.2 pourquoi une fréquence de modulation de 12800Hz ?

✍ *BUP 872 p345*, **Idda**
Pour chiffrer le son, Canal+ module le signal d'origine par une porteuse de fréquence f=12800Hz. Le choix de cette fréquence est due à la plage de fréquence disponible et au matériel utilisé.
Le signal de départ a une dsp qui va jusqu'à 10kHz. (D'ailleurs, à l'époque, la plupart des télévisions ne respectaient pas la norme Hi-Fi qui assure une transmission des signaux jusqu'à 20kHz). Il faut donc f>10kHz pour éviter un repliement spectral autour de 0Hz après modulation pour le chiffrage, et donc perte d'information.
Mais au signal audio chiffré, Canal+ ajoute un signal de fréquence 15625Hz qui, à l'époque des normes Secam (Pal, c'est un peu différent mais c'est la même fréquence), servait à synchroniser l'image. En effet, pour afficher une image, un faisceau d'électron balayait l'écran ligne par ligne et il fallait lui donner le top pour qu'il se mette en début de ligne. Comptez 625 lignes en norme Secam et 25 images par secondes, et on a besoin d'un signal de 625*25=15625Hz. Il faut donc que la dsp du signal après chiffrage n'atteigne pas 15625Hz, sinon on a recouvrement de signaux et perte d'informations.
On a donc besoin d'une fréquence de modulation/démodulation f telle que 10000Hz <f< 15625Hz.

C'est ensuite un choix technologique qui a été fait. On a besoin d'une porteuse stable, facile et peu coûteuse à fabriquer. On a alors utilisé la même technique que les montres : le même quartz suivi de diviseurs de fréquence par deux.
Avec un quartz à 3,2768MHz suivi de 8 diviseurs, on a

$$f = \frac{3.2768 * 10^6}{2^8} = 12800Hz \tag{31}$$

Hors ce quartz est celui utilisé pour les montres allant jusqu'au centième de seconde :

$$\frac{3.2768 * 10^6}{2^{15}} = 100Hz \tag{32}$$

Ce quartz a donc été retenu car très courant (et donc peu cher).

6.3 programme python utilisé

Le code du programme python utilisé pour la construction d'un signal complexe en tant que somme de signaux purs (sinusoïdes) a été perdu. Ce programme avait été téléchargé sur le site de l'agrégation et était très laid. Vous pouvez faire mieux.

voici par exemple un autre programme python téléchargeable sur le site de l'agreg, où l'on construit des triangles à partir de sinusoïdes qui a l'air pas mal. On pourrait ajouter à cela un affichage sur un même graphe des différentes sinusoïdes composant le signal complexe.

```
@author: ENS de Lyon

Triangle Fourier.py

Ce programme calcule et affiche la fonction "triangle" à partir de sa série de Fourier, pour différents or

Les sorties sont des graphes :
- Valeur des a_n fonction de n
- Differentes reconstitution du signal a partir des coefficients de Fourier.
"""


"""
Bibliothèques
"""
import numpy as np
import matplotlib.pyplot as plt

"""
Fonctions de définition des coefficients et du signal théorique
"""
def A(n):   # Amplitude du n-ième terme de la série
    return 1.0/n**2  * 4/np.pi**2 * (np.cos(n*np.pi)-1)

def B(n):   # Amplitude du n-ième terme de la série
    return 0

def triangle(x):     # Triangle théorique (masqué par défaut)
    pi = np.pi
    a = x % (2*pi)
    if a < pi:
        return a/(pi/2)-1
    else:
        return 3-a/(pi/2)


"""
Calcul de la série de Fourier pour les différents ordres
"""
x = np.arange(-4, 8, 0.001);

y1 = 0
for n in range(1,2):
    y1 = y1 + A(n)*np.cos(n*x) + B(n)*np.sin(n*x)    # On somme une à une les composantes jusqu'à N = 1

y3 = 0
for n in range(1,4):
    y3 = y3 + A(n)*np.cos(n*x) + B(n)*np.sin(n*x)    # On somme une à une les composantes jusqu'à N = 3

y6 = 0
for n in range(1,7):
    y6 = y6 + A(n)*np.cos(n*x) + B(n)*np.sin(n*x)    # On somme une à une les composantes jusqu'à N = 6

y10 = 0
for n in range(1,11):
    y10 = y10 + A(n)*np.cos(n*x) + B(n)*np.sin(n*x)    # On somme une à une les composantes jusqu'à N = 10

# Affichage des coefficients de Fourier, puis des fonctions reconstituees
y_triangle = [ triangle(p) for p in x ]
```

```
plt.figure()
plt.plot(np.arange(1,11,1),-A(np.arange(1,11,1)),'o')
plt.xlabel(r'$n$', fontsize=18)
plt.ylabel(r'$a_{n}$', fontsize=18)
plt.show()

plt.figure()
ax1 = plt.subplot(221)
plt.plot(x, y_triangle, "r-")
plt.axis([-4.0,8.0,-1.5,1.5])
plt.plot(x, y1, "b")
plt.title(r'$N=1$', fontsize=18)

ax2 = plt.subplot(222)
plt.plot(x, y_triangle, "r-")
plt.axis([-4.0,8.0,-1.5,1.5])
plt.plot(x, y3, "b")
plt.title(r'$N=3$', fontsize=18)

ax3 = plt.subplot(223)
plt.plot(x, y_triangle, "r-")
plt.axis([-4.0,8.0,-1.5,1.5])
plt.plot(x, y6, "b")
plt.title(r'$N=6$', fontsize=18)

ax4 = plt.subplot(224)
plt.plot(x, y_triangle, "r-")
plt.axis([-4.0,8.0,-1.5,1.5])
plt.plot(x, y10, "b")
plt.title(r'$N=10$', fontsize=18)

plt.show()
```

Place vide non utilisée immédiatement car retard du poly