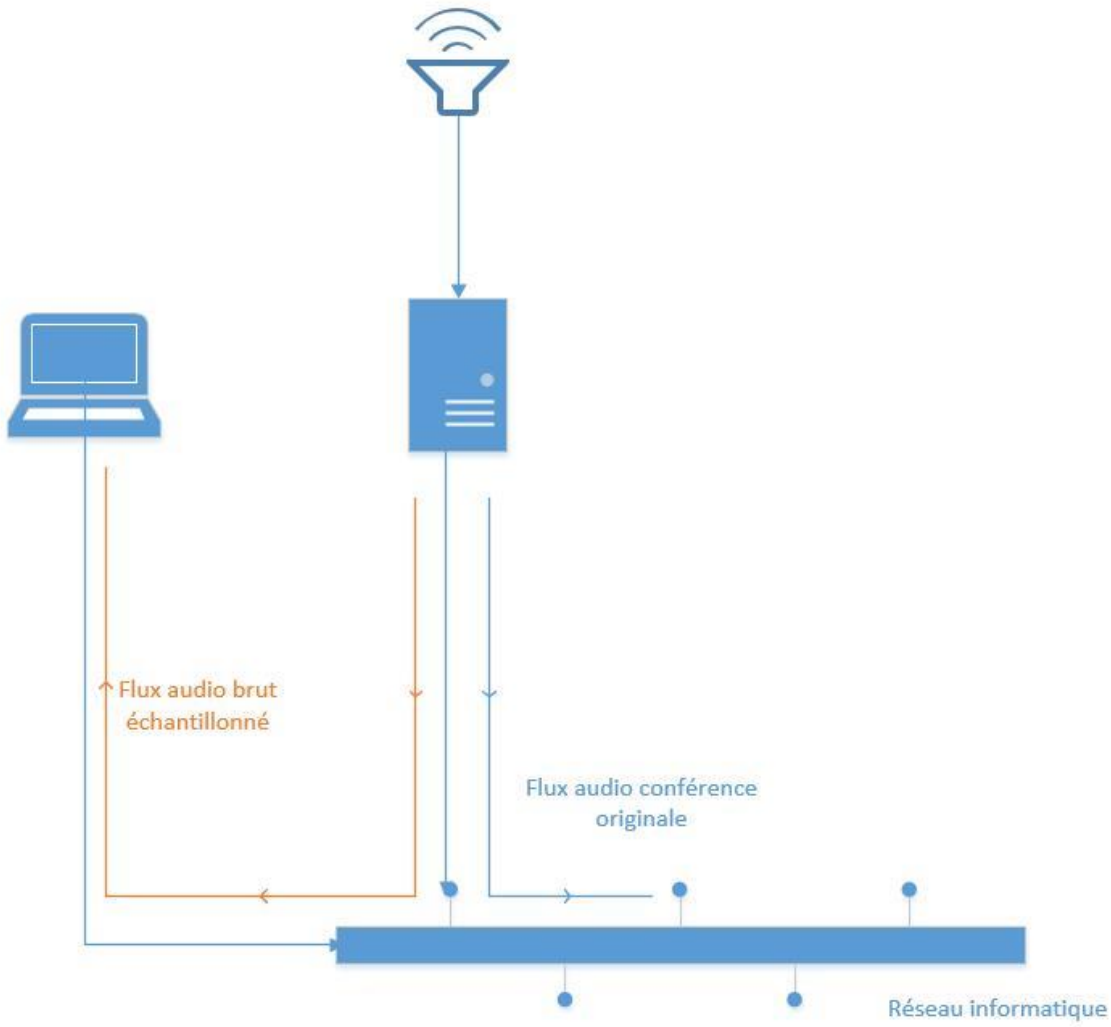
Coffin Mattéo
Cormier Mathieu

Lucas Paul
Petibois Benjamin

LIVRABLE 2 : FILTRE



SOMMAIRE :

Coffin Mattéo
Cormier Mathieu

Lucas Paul
Petibois Benjamin

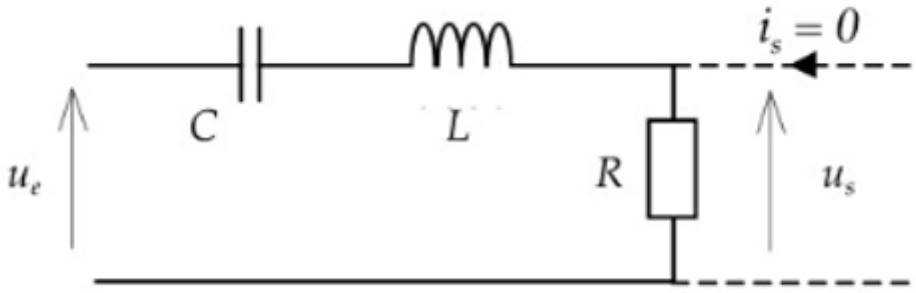
1) Calcul des valeurs de R, L et C correspondant au montage électronique théorique.

Nous allons donc chercher un intervalle de fréquence qui nous intéresse, nous utiliserons donc des filtres passe-bandes permettant d'isoler nos fréquences recherchées avec leurs bandes de fréquences et éliminer toutes les autres fréquences.

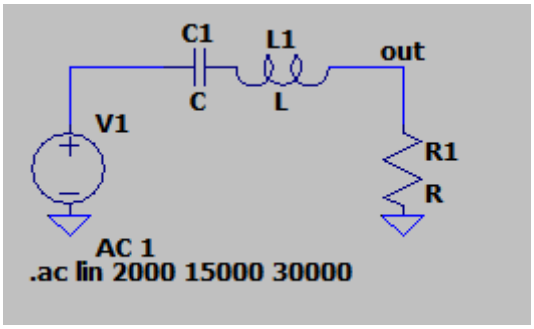
Nous avons choisi au livrable 1, 5 signaux porteurs modulé qui ont pour fréquences respectives propres : 18, 19, 20, 21 et22 KHertz. Ce sont leurs modulations qui vont permettre le transfert de l'information. Et donc il faudra prendre en compte ces modulations dans nos filtres. Mais ces modulations ne devront pas trop modifier la fréquence propre de l'onde porteuse. Sinon notre bande passante devra être plus large et donc serait plus susceptible de contenir des sons non désirés. C'est pour cela que nous allons considérer que la modulation de fréquence doit être 65 fois inférieur à la fréquence propre de l'onde. Ce sera note facteur qualité du filtre.

Donc pour la suite, on prendra un facteur qualité de 65. Nous prendrons donc pour isoler cette bande passante, pour chaque onde porteuse, un filtre passe-bande de facteur qualité 65, dont les valeurs R, L et C vont changer car elles vont dépendre de la fréquence propre ou de la porteuse.

On peut faire cela en utilisant un filtre passe bande RLC de la forme :



Sur LTspice, le shéma du montage électrique équivalent au filtre réalisé est présenté comme cela :



Coffin Mattéo
Cormier Mathieu
Lucas Paul
Petibois Benjamin

Avec R la valeur de résisance en ohm, C la capacité du condensateur en farades et L l'inductance de la bobine en Henry.

Dans ce type de filtre, , on peut dans un premier temps déterminer sa fonction de transfert noté $H(j\omega)$ qui est égal à la tension de sortie Us divisé par la tension d'entrée Ue :

$H(j\omega)= Us/ Ue$

Grace au pont diviseur de tension, on peut en déduire l'égalité suivante :

$$Us=\frac{Z_R}{Z_R+Z_C+Z_L} Ue$$

Donc : $H(j\omega)=\frac{Z_R}{Z_R+Z_C+Z_L}$

Comme on sait que :

$$Z_R = R \;\; Z_C \;= \frac{1}{jC\omega} \;\; Z_L = jL\omega$$

Alors on en déduit que : $H(j\omega)=\frac{R}{R+jL\omega+\frac{1}{jC\omega}} = \frac{jRC\omega}{1-LC\omega^2+jRC\omega}$

Sachant que la pulsation propre du circuit s'exprime par :

$$\omega_0 = \frac{1}{\sqrt{LC}}$$

Avec le facteur de qualité Q :

$$Q = \frac{1}{RC\omega_0} = \frac{1}{R} \times \sqrt{\frac{L}{C}}$$

Et la fréquence propre f_0 s'exprime par :

$$f_0 = \frac{\omega_0}{2\pi} = \frac{1}{2\pi\sqrt{LC}}$$

Alors la fonction de transfert peut s'écrire en fonction de la pulsation propre telle que :

$H(j\omega)=\frac{j\frac{\omega}{Q\omega_0}}{1-(\frac{\omega}{\omega_0})^2+j\frac{\omega}{Q\omega_0}}$

Le gain en décibel s'exprime par : $G_{dB} = 20\log\left(\frac{\omega}{Q\omega_0}\right) - 10\log\left(\left[1-\left(\frac{\omega}{\omega_0}\right)^2\right]^2 + \left(\frac{\omega}{Q\omega_0}\right)^2\right)$

Et la phase s'exprime par : $\varphi = \frac{\pi}{2} - \arctan\frac{\frac{\omega}{Q\omega_0}}{1-\left(\frac{\omega}{\omega_0}\right)^2}$

En faisant l'étude théorique, on se rend compte que :

Coffin Mattéo
Cormier Mathieu

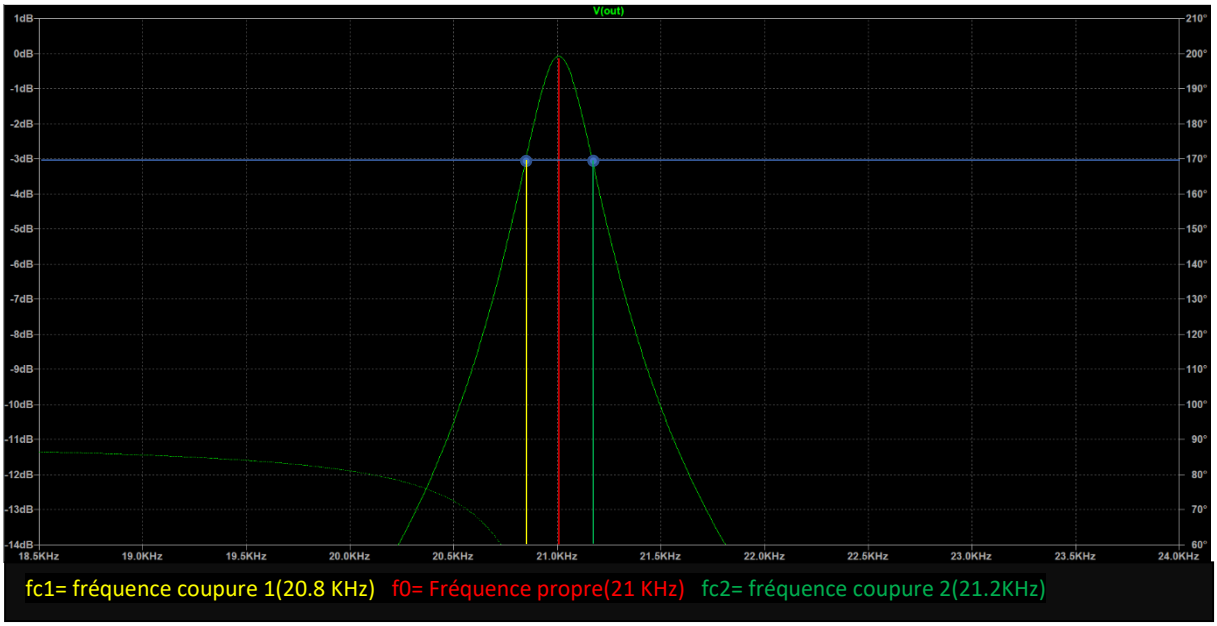
Lucas Paul
Petibois Benjamin

Si $\omega \ll \omega_0$, $\frac{\omega}{\omega_0} \ll 1$ alors $G_{dB} \rightarrow 20\log\left(\frac{\omega}{\omega o}\right) - 20\log Q$.
On a une droite de pente 20dB/décade.

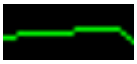
Si $\omega = \omega_0$, alors $G_{dB} = 0$.

Si $\omega \gg \omega_0$, $\frac{\omega}{\omega_0} \gg 1$ alors $G_{dB} \rightarrow -20\log\left(\frac{\omega}{\omega o}\right) - 20\log Q$.
On a une droite de pente -20dB/décade.

Si on devait donc représenter l'évolution du gain et de la phase dans un filtre de ce type on obtient :



Légende :

 = Courbe Gain en dB/fréquence

 = Courbe phase en °/fréquence

Prenons l'exemple de la fréquence porteuse de 21000 Hertz.

Il faut donc dans un premier temps adapter les valeurs de C et L pour que la fréquence propre (pour laquelle le gain est égal à 0) soit égale à 21000 Hertz . On pose donc :

$$f_0 = 21000 = \frac{1}{2\pi\sqrt{LC}}$$

On va donc choisir arbitrairement une valeur correcte pour le condensateur comme 1µF. On peut maintenant isoler L :

$$L = \frac{1}{(21000 \times 2\pi)^2 \times C} = 5.74 \times 10^{-5} H$$

On peut faire la même chose si on choisit de définir d'abord L :

$$C = \frac{1}{(f_0 \times 2\pi)^2 \times L}$$

Maintenait que nous avons les valeurs de L et C, il nous reste à déterminer la valeur R.

La valeur R dépend de Q. On le voit grâce à l'équation :

$$\frac{1}{R} = \frac{Q}{\sqrt{\frac{L}{C}}}$$

$$R = \frac{\sqrt{\frac{L}{C}}}{Q}$$

Donc comme la bande passante (noté Δf) se vérifie par l'égalité suivante : $\frac{f_0}{Q}$ alors on peut en déduire quelle sera la borne passante pour chaque onde porteuse à partir de notre facteur qualité.

Donc pour notre exemple $\Delta f = \frac{21000}{65} = 307.7\ Herts$

Donc notre bande passante sera de 307.7 Hz. On en déduit donc les fréquences de coupures théoriques notés $f_1\ et\ f_2$:

$f_1 = f_0 - \frac{\Delta f}{2}$

$f_2 = f_0 + \frac{\Delta f}{2}$

Ainsi, on applique cette méthode pour toutes nos ondes porteuses. On obtient à la fin toutes les valeurs R, L et C pour filtrer chaque fréquence que l'on souhaite.

Pour nous faciliter la tâche, nous avons préparé un code python afin de réaliser nos calculs pour la valeur f_0 à définir et les valeurs Q et C prédéfinis :

Coffin Mattéo
Cormier Mathieu

Lucas Paul
Petibois Benjamin

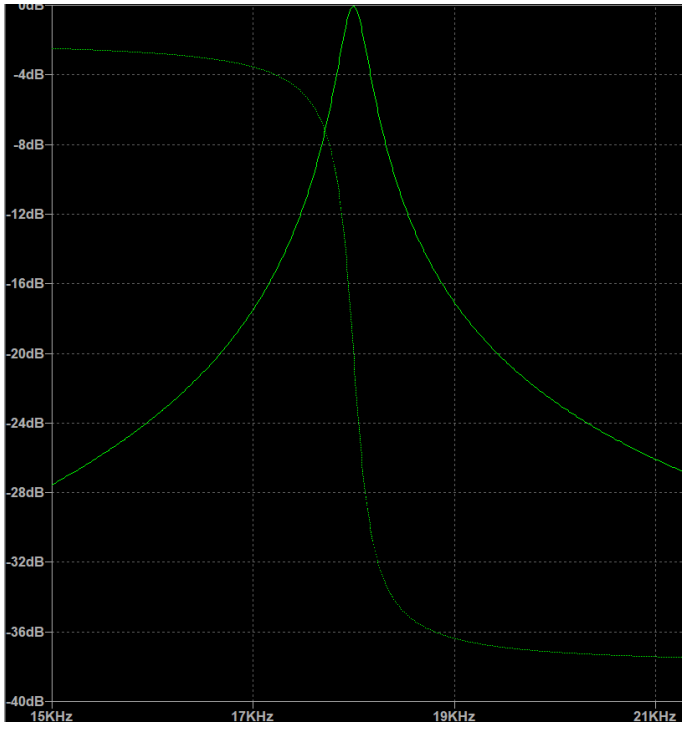
```
1
2
3   from math import *
4
5
6   C = 1
7   C = C*(10**-6)
8   F = int(input("Fréquence propre"))
9
10
11  L = (1/(((F*2*pi)**2)*C))
12  Q = 65
13  R = (sqrt(L/C))/Q
14  print('Le condensateur à pour valeur',C,' Henry')
15  print('La bobine à pour valeur', L,' Henry')
16  print('La résistance à pour valeur',R,' ohm')
17
```

Il est possible aussi d'utiliser un seul filtre pour filtrer toutes les fréquences porteuses avec leurs modulations. Mais pour cela, il faudra que notre bande de fréquence soit bien plus large. Ce qui implique un facteur qualité plus petit. Mais le souci, c'est que les sons parasites risquent d'être nombreux puisque qu'on va prendre en compte des fréquences qui ne nous intéressent pas. Ce qui risque d'être problématique.

2) Fréquence(s) de coupure et diagramme de Bode correspondant au filtre désiré :

Ainsi, avec notre méthode, on a pu réaliser un filtre passe bande pour chaque fréquence porteuse :

Diagramme de Bode RLC (filtre passe bande pour 18kHz)



Coffin Mattéo
Cormier Mathieu

Lucas Paul
Petibois Benjamin

On constate que pour nos calculs de composant pour un seuil de filtrage passe bande (18KHz) est justifié par notre diagramme de Bode.

*Autres Filtres à mentionné (utiliser pour le projet) :

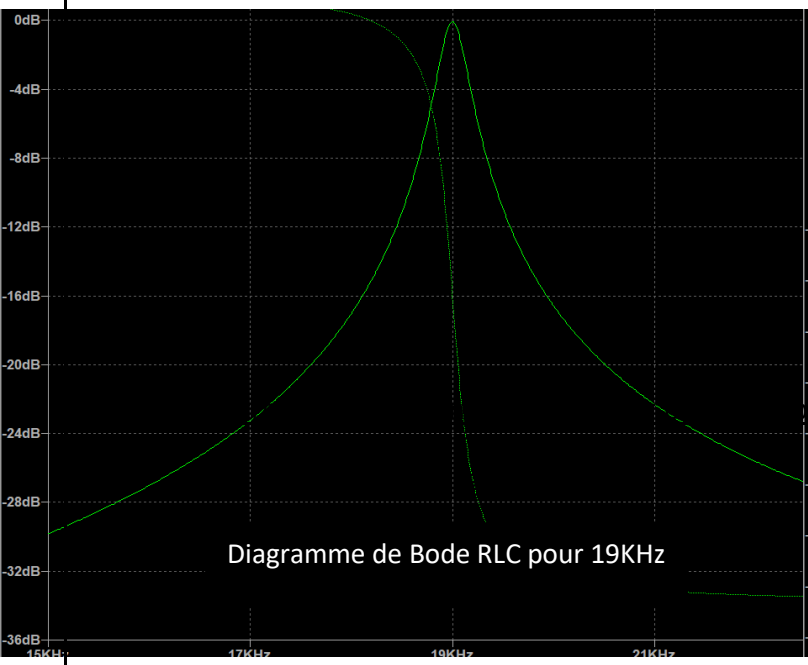


Diagramme de Bode RLC pour 19KHz

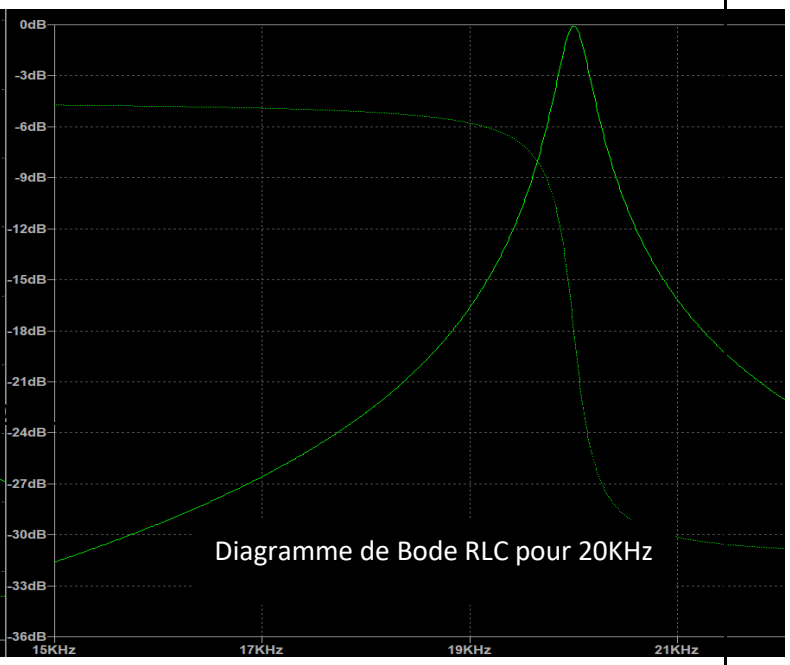


Diagramme de Bode RLC pour 20KHz

Légende :

= Courbe Gain en dB/fréquence

= Courbe phase en °/fréquence

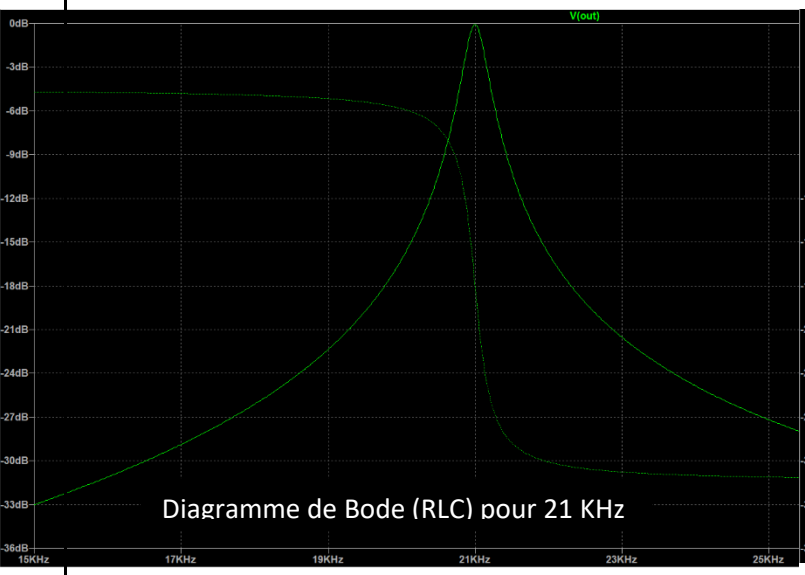


Diagramme de Bode (RLC) pour 21 KHz

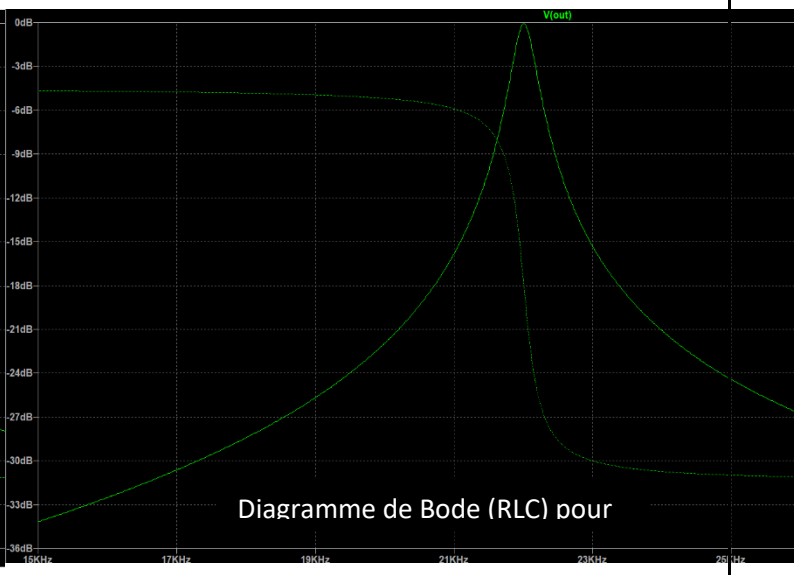


Diagramme de Bode (RLC) pour

Coffin Mattéo
Cormier Mathieu

Lucas Paul
Petibois Benjamin

3) Code Python permettant de réaliser le filtrage numérique du signal audio échantillonné :

Code d'un des filtres :

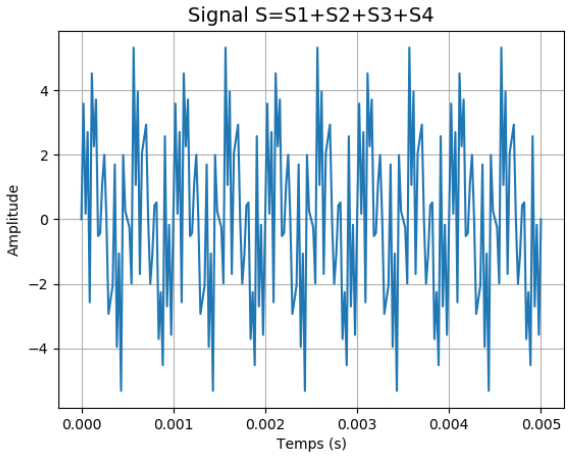
```
from numpy import *                     #import de numpy
from scipy.signal import periodogram    #import du module signal de scipy
from pylab import *                     #import de matplotlib.pylab

Fe=45000                    #fréquence d'échantillonnage
D= 10                         #Durée du signal en seconde
t = arange(0, 0.005, 1/Fe)        #creation de la base temps avec numpy
f1= 2000                          #fréquence du signal S1
f2= 18000                         #fréquence du signal S2
f3 = 20000
f4 = 9000

# S1 et S2 et S=S1+S2
S1 = 2*sin(2*pi*f1*t)                  #creation d'une sinusoide de Fréquence f1
S2 = sin(2*pi*f2*t)
S3 = 2*sin(2*pi*f3*t)
S4 = 2*sin(2*pi*f4*t)
  #creation d'une sinusoide de Fréquence f2
S= S1+S2+S3+S4


#calcul de la transformée de Fourier avec la fonction periodogram

f,FFT = periodogram(S,Fe)               # f: vecteur des fréquences et FFT:la transformée de Fourier du signal S=S1=S2=S3=S4


# Affichage du signal-----------------------------------------------------'
fig=figure(0)
plot(t,S)                       #Affichage via la fonction plot de Matplotlib
xlabel('Temps (s)')           #définition de l'axe des abscisses
ylabel('Amplitude')           #définition de l'axe des ordonnées
plt.title ('Signal S=S1+S2+S3+S4',fontsize=14)
plt.grid()
show()                          #affichage des courbes


# Affichage du de la transformée de Fourier FFT du signal S=S1=S2=S3=S4-----------------

plot(f,FFT)                     #Affichage via la fonction plot de Matplotlib
xlabel('Fréquence (Hz)')        #définition de l'axe des abscisses
ylabel('Module')                #définition de l'axe des ordonnées
plt.xlim(0, 22000)
plt.grid()
plt.title ('Module de la Transformée de Fourier du signal S=S1+S2+S3+S4',fontsize=14)
show()
```

Cette partie de code permet définir les librairies, les variables, la transformation de Fourier du signal (f,FFT = periodogram(S,Fe))et de les illustrés avec nos courbes.

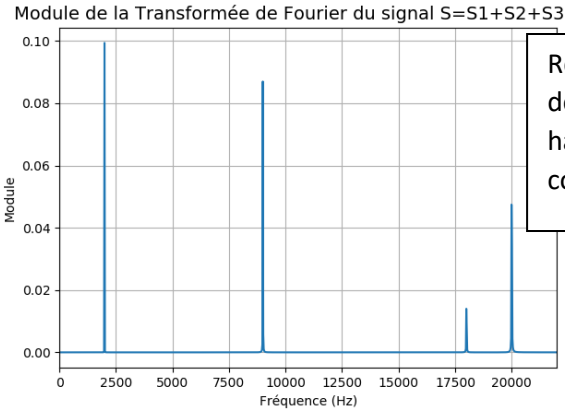
Les sinusoïdales sont créées à partir de l'équation $\sin(2 \times pi \times f \times t)$ ainsi notre équation prendra la valeur en abscisse de la fréquence choisie.

Coffin Mattéo
Cormier Mathieu

Lucas Paul
Petibois Benjamin



Signaux combinés permettant de faire une simulation du filtre passe bande



Représentation de la Transformation de Fourier. Représentation des harmoniques présents dans la conférence.

Coffin Mattéo
Cormier Mathieu

Lucas Paul
Petibois Benjamin

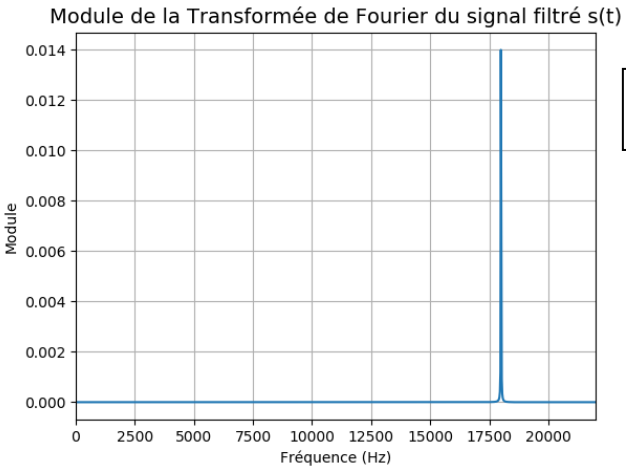
```
#Définir la fréquence de coupure fc filtre à 17500 Hz
#Définir la fréquence de coupure fb filtre à 18500 Hz
fc= 17500
fb = 18500
# On définit une variable qui reçoit le signal filtré de la même taille que la transfomrée de fourier(FFT)
FFT_filtre= FFT


#On réalise un filre passe bas comme suit:

for i in range(len(f)):
    if f [i] < fc: # on coupe toutes les fréquences < 17500 Hz
        FFT_filtre [i] = 0.0
for i in range(len(f)):
    if f [i] > fb: # on coupe toutes les fréquences > 18500 Hz
        FFT_filtre [i] = 0.0

#On calcul la transfomrée de Fourier inverse du signal après filtrage en utilsant la focntion ifft de Python
# la FFT inverse permet de revenir dans l'espace temporal (espace Fourier -> espace temps)
FFT_inverse = np.fft.ifft(FFT_filtre)

fig=figure(0)
plot(f,FFT_filtre)              #Affichage via la fonction plot de Matplotlib
xlabel('Fréquence (Hz)')        #définition de l'axe des abscisses
ylabel('Module')                #définition de l'axe des ordonnées
plt.xlim(0, 22000)
plt.grid()
plt.title ('Module de la Transformée de Fourier du signal filtré s(t)',fontsize=14)
show()

plot(FFT_inverse)               #Affichage via la fonction plot de Matplotlib

ylabel('Amplitude')                #définition de l'axe des ordonnées
plt.xlim(0, 50)
plt.grid()
plt.title ('Le signal obtenu après filtrage',fontsize=14)
show()
```

Ici cette partie comprend le codage du filtre sur python, ainsi les valeurs utilisées pour le filtre passe bande sont 17 500 hertz et 18 500 hertz qui dépendra de l'indice de qualité.

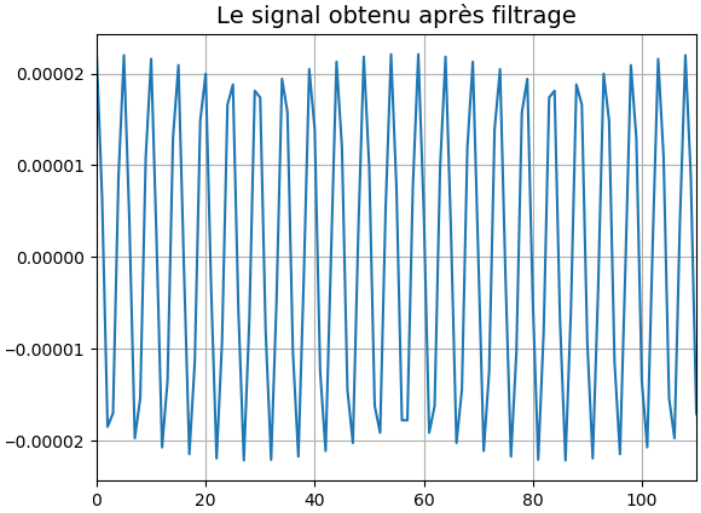
Ici le filtrage est informatique soit l'utilisation d'une boucle for et d'un décompte range permettant d'avoir toute la plage de notre fréquence. Ainsi par les conditions de notre code si la fréquence est inférieure à notre fréquence de coupure celle-ci sera mise à 0.0 (éliminer).

Par l'utilisation de la librairie de Fourier, nous pouvons illustrer nos courbes avec notre ou nos filtres.

Coffin Mattéo
Cormier Mathieu
Lucas Paul
Petibois Benjamin



Signal initial après filtrage



Nous avons donc après filtrage obtenue une seule harmonie et son signal, on constate donc un signal périodique sinusoïdale.

Conclusion :

Nous aurons donc 5 filtres passe bande. Elles auront des fréquences de coupures prédéfinies (entre 18, 19, 20, 21 et 22KHz) variant avec l'indice de qualité. Chaque filtre est utilisé pour isoler l'onde à traiter. Ainsi nous pourrons traiter nos informations cas par cas sans perturbations (voix, parisite…).

Ainsi, l'utilisation du code de filtre Python diffère de nos filtres électroniques. En effet les filtres électroniques varient selon les valeurs du condensateur, de la bobine et de la résistance, contrairement à notre code qui lui fait varier la fréquence à zéro si celle-ci ne convient pas.