Cours : Traitement du signal **TP n°3 : Filtering of signals**	ISEN ALL IS DIGITAL! BREST / yncréa	Houssem Gazzah CIPA4 Nantes

	Nom MARQUET *Prenom Félix*

The objective of this experiment is to learn about noise-corrupted signals and ways to combat noise
Noise is any unwanted signal that adds to the signal of interest. There are two types of noise:
Ambient noise: is the superposition of all unwanted signals that unintentionally reach the receiver.
Mainly, it includes emissions from surrounding equipments, cosmic noise, and noise generated by the receiver itself. The result is a low-power signal with flat infinite (in practice very large) spectrum.
Jamming noise: is a signal that is transmitted by a third party in order to corrupt the signal of interest.
As a modulated signal, it has a narrow-band spectrum centered at some frequency
To achieve jamming, this central frequency should be in the bandwidth of the signal to be corrupted.
For the jamming to be effective, the jamming signal needs to have sufficient power.

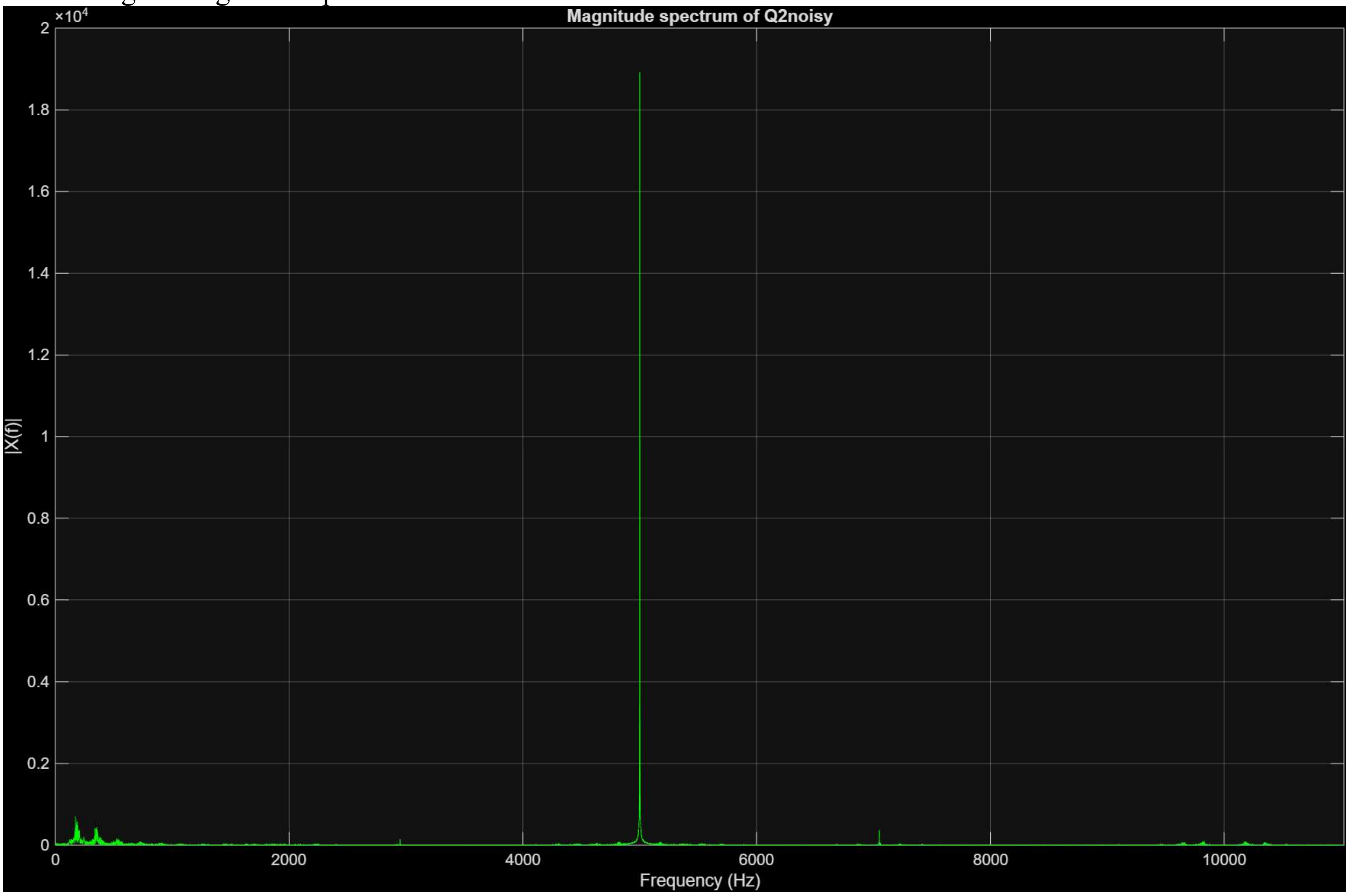
Any noise located outside the signal bandwidth can be removed by means of filters, which are LTI systems.
MATLAB imitates discrete-time LTI system characterized by z-transforms of the form B(z)/A(z).
Many types of filters exist, among which Butterworth and Elliptic filters are designed in Matlab using functions BUTTER and ELLIP that allow one to obtain (the coefficients of) polynomials A and B.

An ideal filter has a square-shaped frequency response i.e. 0 or 1 response
In practice, this can only be approximated. A good approximation requires a long filter, i.e. a high order **N**
Frequencies need to be normalized w,r,t the sampling rate before being passed to BUTTER
[B,A]=BUTTER(N,0.7) → low-pass with cutoff at 0.7 times fs/2
[B,A]=BUTTER(N,[0.5 0.8]) → pass-band with passband [0.5,0.7] times fs/2
[B,A]=BUTTER(N,0.7,'high') → high-pass with cutoff at 0.7 times fs/2
[B,A]=BUTTER(N, [0.5 0.8],'stop') → stop-band with stopband [0.5,0.7] times fs/2
Once obtained, the filter (i.e. A and B parameters) is used to process data following
outputSIGNAL = FILTER(B,A,inputSIGNAL)

Experiment 1

The objective of this experiment is to extract a narrow-band noise from a corrupted signal by means of a properly designed filter

Load the WAV file **Q2noisy**
It corresponds to an original audio signal corrupted by additive and jamming noise signals.
Plot the signal magnitude spectrum. **Insert here.**



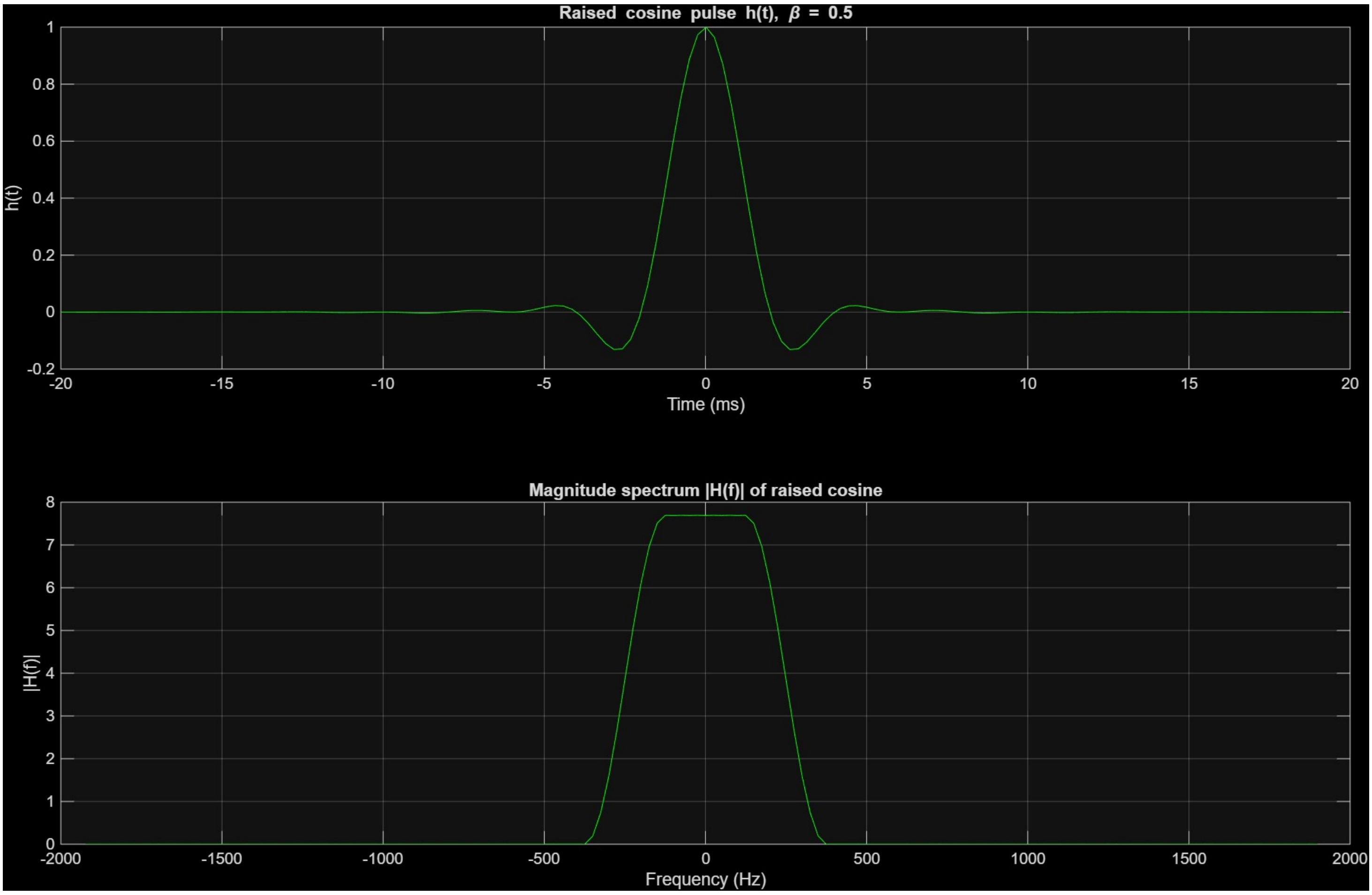
Jamming signal at frequency 5kHz
Design a suitable filter and process the corrupted signal. Save the result in **E71yourname**

Experiment 2

Generating the signal of interest
Generate the following raised cosine *h(t)* over [-10T,10T], where

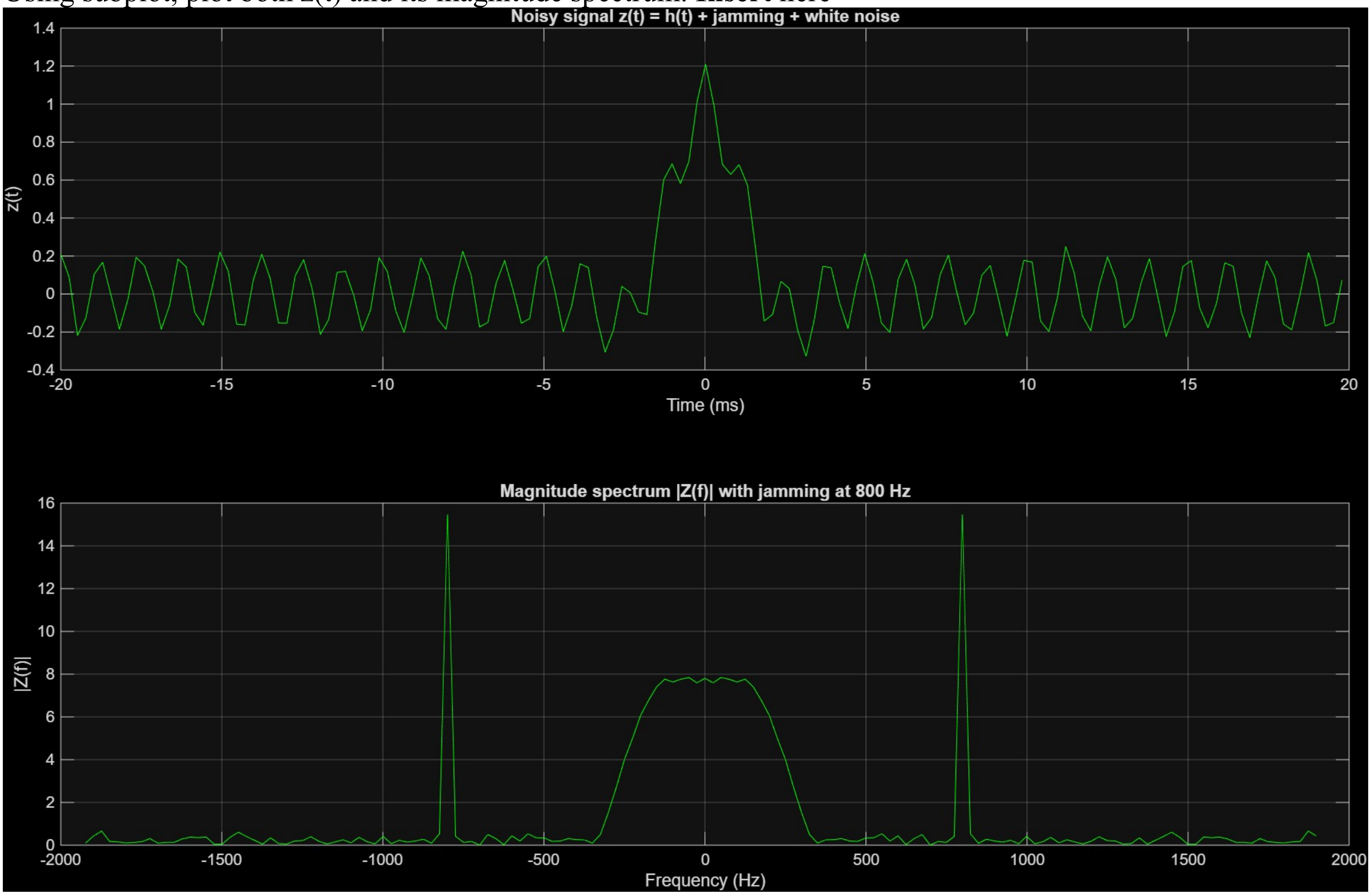
$$h(t) = \mathrm{sinc}\left(\frac{t}{T}\right)\frac{\cos\left(\frac{\pi\beta t}{T}\right)}{1-\frac{4\beta^2t^2}{T^2}}$$

T=2 ms and β=0.5. Consider a sampling period of *0.13 T*
Using subplot, plot both h(t) and its magnitude spectrum. Insert here



Generation of jamming (narrowband) noise: Generate a sinusoidal signals x(t) at 800 Hz
Generation of ambient (white) noise: Generate low-level noise n(t) using randn

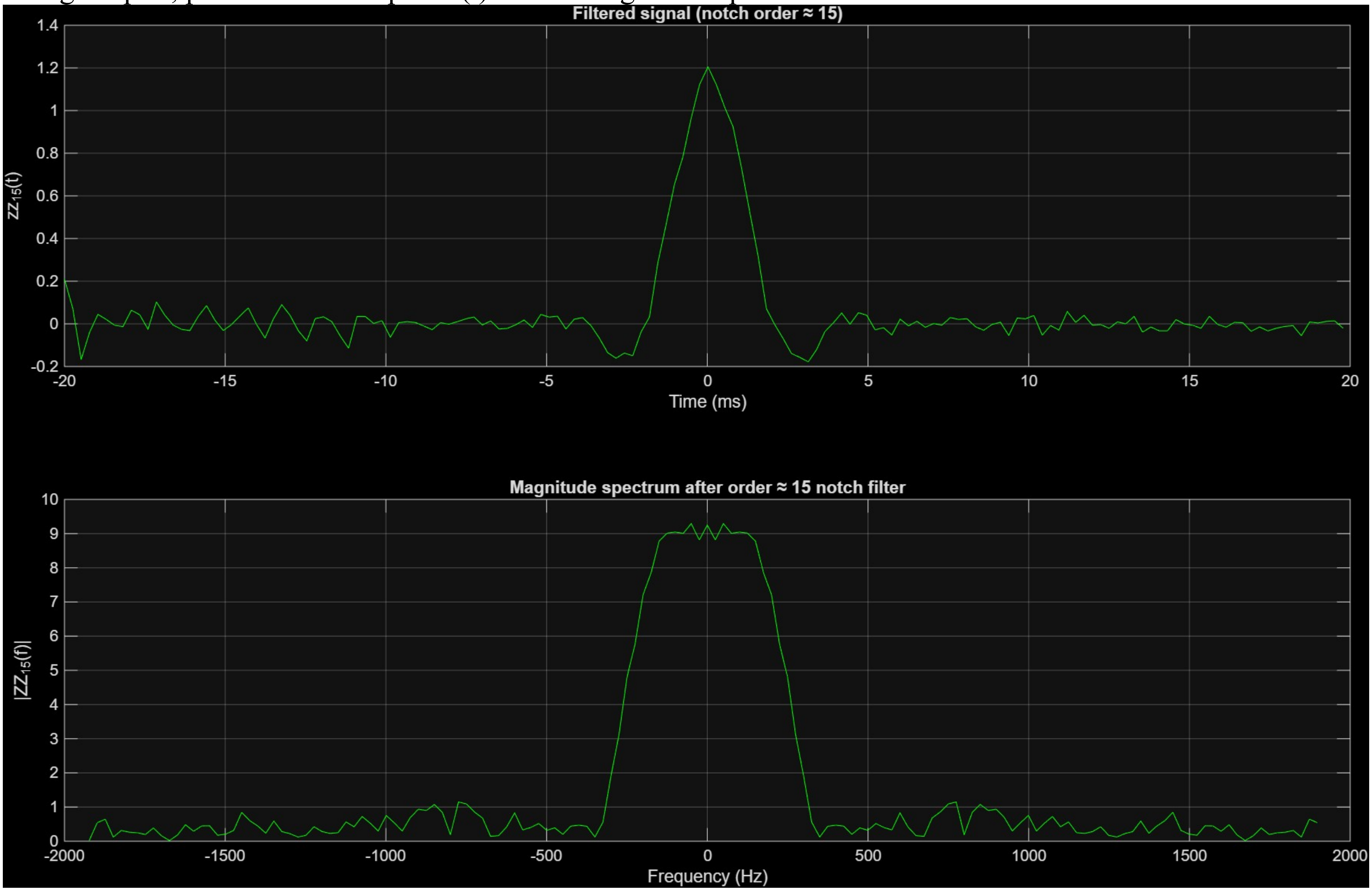
Set $z(t)= h(t)+ x(t)+ n(t)$
Using subplot, plot both z(t) and its magnitude spectrum. **Insert** here



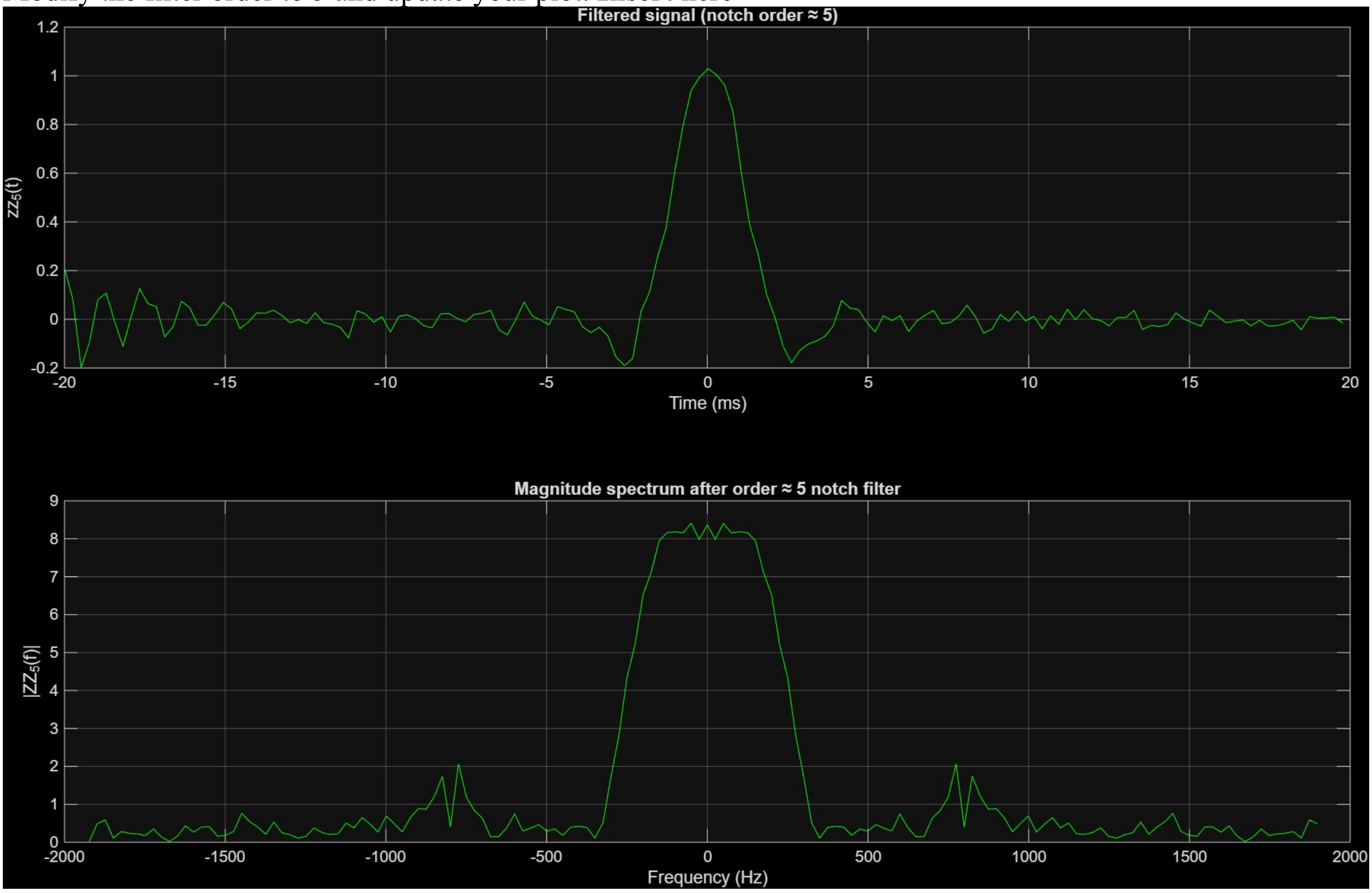
Filter design

Design an appropriate 15-th filter to remove the jamming signal

Using subplot, plot the filter output zz(t) and its magnitude spectrum. **Insert** here

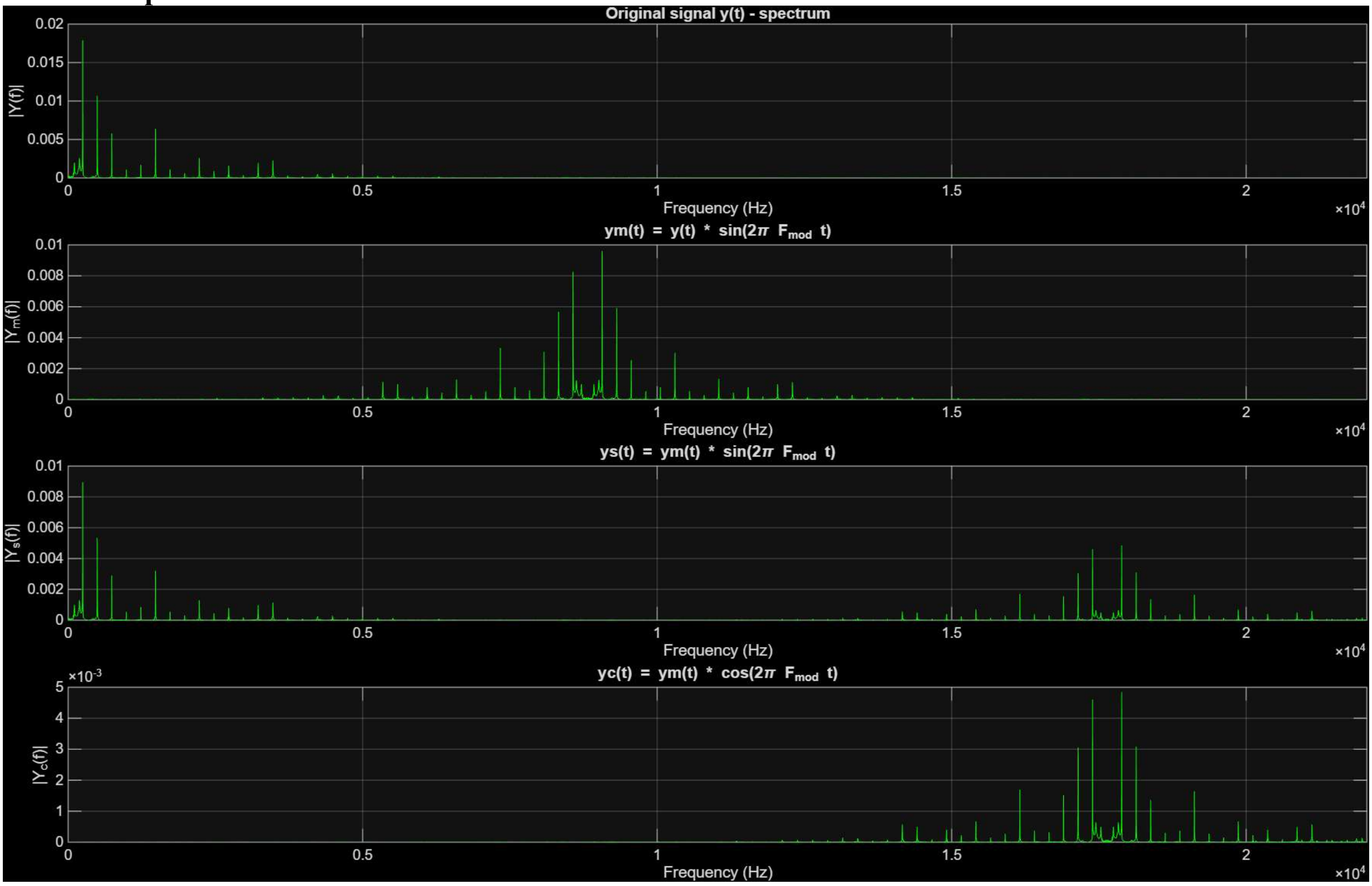


Modify the filter order to 5 and update your plot. **Insert** here



Experience 3

Charger le signal contenu dans le fichier guitar.wav dans un vecteur y
Fixer Fmod=8820 Hz
Générer 3 signaux qui correspondent respectivement à
$y_m(t)=y(t)\sin(2\pi.Fmod.t)$
$y_s(t)=y_m(t)\sin(2\pi.Fmod.t)$
$y_c(t)=y_m(t)\cos(2\pi.Fmod.t)$
Obtenir et visualiser le spectre d'amplitude de chacun des signaux y(t), $y_m(t)$,$y_s(t)$, $y_c(t)$
Insérer les plots ici



Justifier l'allure des spectres en se référant aux propriétés de la transformée de Fourier
$y_m(t)$ modulation DSB-SC → le spectre de $y(t)$ est translaté autour de $\pm F_{\text{mod}}$
$y_s(t)$ re-multiplication par sin → produit de deux sinusoïdes, d'où un terme à basse fréquence (proche de $y(t)$) et un terme image autour de $2F_{\text{mod}}$.
$y_c(t)$ re-multiplication par cos → spectre centré entre 0 et $2F_{\text{mod}}$, mais sans composante continue pure, ce qui rend la récupération plus délicate par simple filtrage passe-bas.

Ecouter les sons correspondants à $y_m(t)$,$y_s(t)$, $y_c(t)$
En déduire lequel des trois signaux peut être manipulé pour récupérer le son original de y(t) ?
Proposer et exécuter une telle manipulation, dite démodulation
Décrire cette manipulation ici…..
On choisit $ys(t)$ comme signal à démoduler.
On remonte la théorie : $ys(t) = ym(t)sin\,(2\pi F_{mod}t) = y(t)sin\,^2(2\pi F_{mod}t) = \frac{1}{2}y(t) - \frac{1}{2}y(t)cos\,(4\pi F_{mod}t)$.
Le spectre de $ys(t)$ contient donc une copie basse fréquence du signal original (multipliée par 1/2) et une copie déplacée autour de $2F_{\text{mod}}$.
La démodulation consiste à appliquer un filtre passe-bas sur $ys(t)$ pour supprimer la partie haute fréquence autour de $2F_{\text{mod}}$ et ne garder que le terme $\frac{1}{2}y(t)$. On obtient alors une version filtrée $y_{\text{rec}}(t)$ proportionnelle à $y(t)$, que l'on peut amplifier (gain ≈ 2) pour retrouver le niveau d'origine.

Obtenir et visualiser le spectre d’amplitude de ce signal

Insérer ici