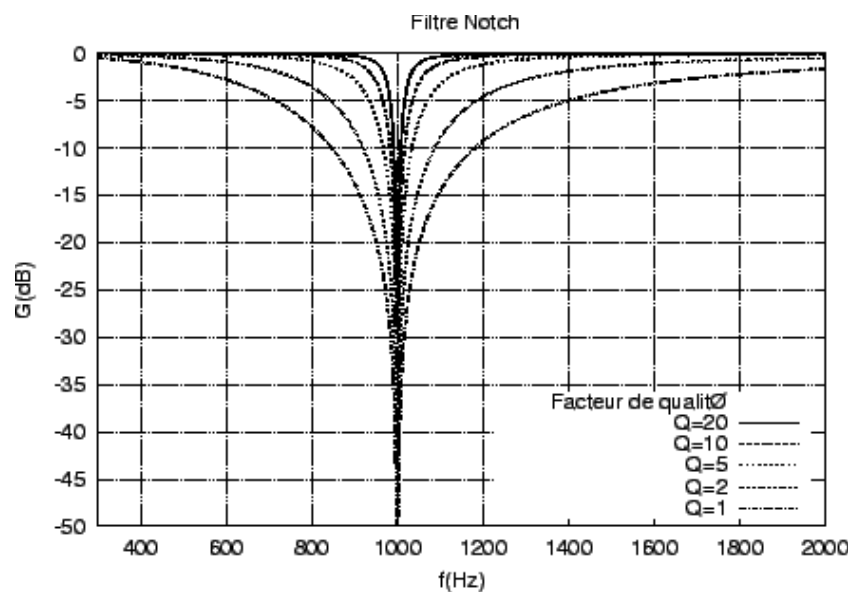


Rapport de projet

Filtrage numérique d'un enregistrement audio



Cyril Royer

3^e année IMACS
INSA Toulouse

Rapport daté du 13/05/2017

Introduction

Ce rapport s'inscrit dans le cadre de la matière « Filtrage Numérique » de la 3^è année IMACS de l'INSA de Toulouse. Il décrit l'ensemble des raisonnements et démarches suivis afin de résoudre un problème de filtrage numérique. En effet, le besoin en filtrage numérique est omniprésent de nos jours : Télécommunication, traitement de musique ou d'images, annulation d'écho, etc.

Nous analyserons dans un premier temps le signal à traiter. Puis, au vu du problème, nous listerons les solutions possibles. Enfin, nous tenterons de mettre en place ces solutions.

Note : Ce rapport contient des liens hypertexte. Ces liens vous permettront de télécharger les fichiers audio ainsi que les scripts Matlab relatifs à ce projet.

Tous les fichiers sont disponibles sur <https://www.cyrilroyer.com/telechargements.html>

Sommaire

I) Analyse du signal à traiter	3
II) Pistes de solutions	4
1) Passage en fréquentiel puis retour en temporel	4
2) Conception d'un filtre numérique de A à Z	4
3) Utilisation d'un filtre existant adapté au problème	4
III) Mise en place des solutions	5
1) Filtre 'maison'	5
2) Filtre Notch	7
Conclusion	8

I) Analyse du signal à traiter :

Le signal à traiter est un [enregistrement vocal d'un 'bonjour'](#). La voix est malheureusement presque entièrement masquée par un bip très désagréable.

Nous n'en saurons pas davantage simplement en écoutant l'enregistrement. On procède donc à son analyse fréquentielle (FFT). Nous utilisons pour cela le logiciel open-source Audacity. Nous utilisons une FFT à 1024 points seulement afin de mieux visualiser le problème.

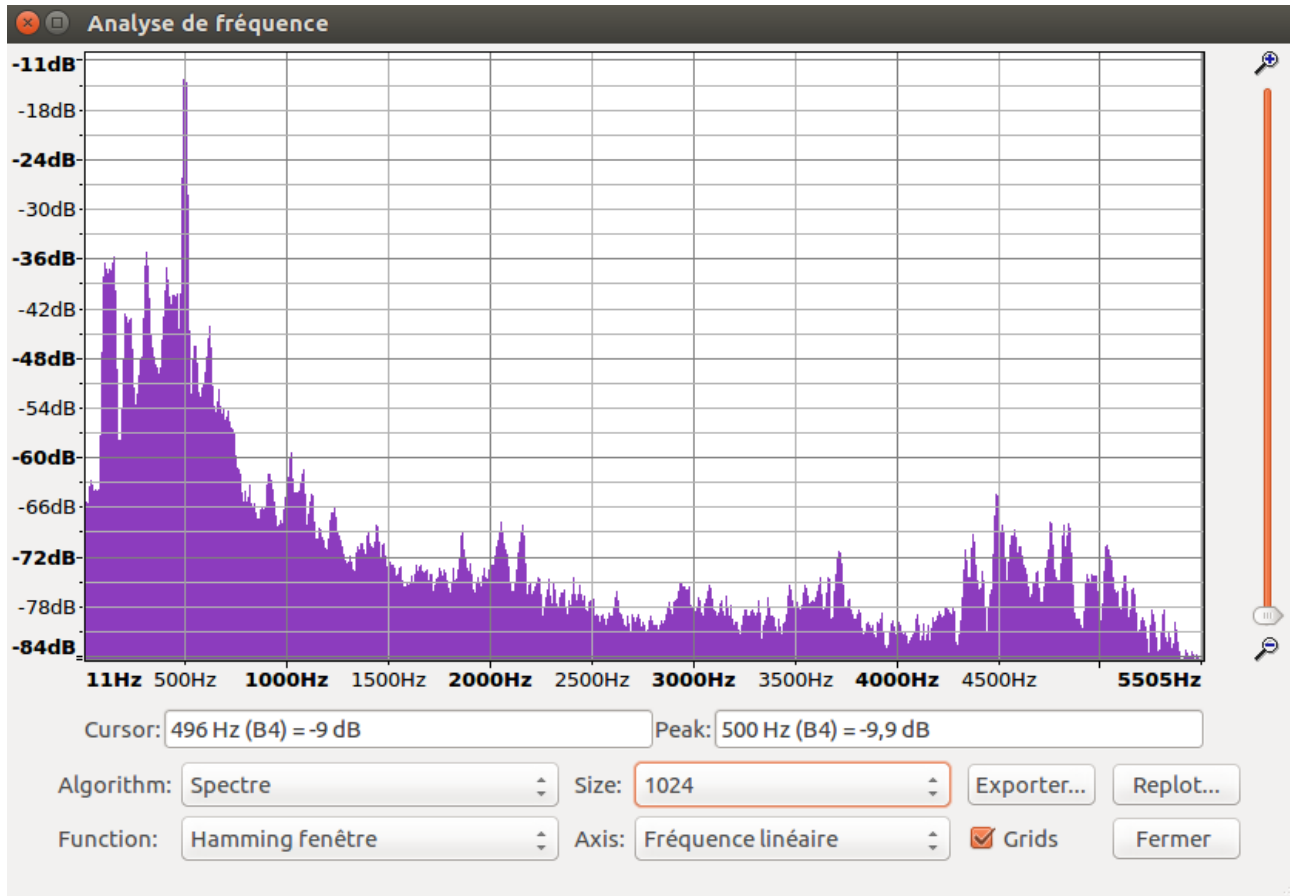


Figure 1 : Tracé du spectre original sous Audacity

On observe aux alentours de 500Hz un pic d'amplitude -9dB, soit près de 40dB de plus que les autres pics de fréquence. Ce pic, situé en plein milieu du spectre audible, est clairement à l'origine du problème. Voyons maintenant quelles sont les solutions pour supprimer ce pic de fréquence du signal.

II) Pistes de solutions

Cette partie décrit quelles sont nos options pour filtrer ce signal.

1) Passage en fréquentiel puis retour en temporel

La première solution qui nous vient à l'esprit est à la fois la plus simple et la plus efficace. Elle consisterait simplement à effectuer la transformée de Fourier du signal, supprimer ou atténuer la fréquence de 500Hz, puis effectuer la transformée de Fourier inverse du signal. Nous ne tenterons pas cette solution, car la transformée de Fourier est une opération lourde et coûteuse en temps processeur. Et c'est bien le but des filtres numériques que d'éviter cette transformée de Fourier.

2) Conception d'un filtre numérique de A à Z

La solution la plus élégante serait de suivre la méthode décrite dans le cours. Cette méthode consiste à concevoir le filtre en continu, puis d'effectuer la transformée bilinéaire de façon à obtenir les coefficients du filtre numérique. Malgré sa difficulté, nous tenterons cette méthode.

3) Utilisation d'un filtre existant adapté au problème

La troisième méthode serait de simplement utiliser une fonction de filtrage présente dans Matlab de façon à obtenir les coefficients du filtre. Cette méthode est à la fois simple, efficace, et économe en ressources. Ce sera la solution retenue au final.

III) Mise en place des solutions

On cherche à mettre en place les solutions décrites dans la partie 2.

1) Filtre 'maison'

Essayons dans un premier temps de concevoir un filtre numérique capable de supprimer le pic de fréquence. Cela passe d'abord par la conception d'un filtre continu. Les seuls filtres continus que nous connaissions sont les filtres passe-haut/passe-bas. Ces filtres présentent une seule fréquence de coupure, ainsi qu'une pente de -20dB/décade , ce qui est largement insuffisant pour notre problème. Il peuvent cependant être combinés de façon à augmenter cette pente et multiplier les fréquences de coupure.

La forme générale du filtre souhaité est la suivante :

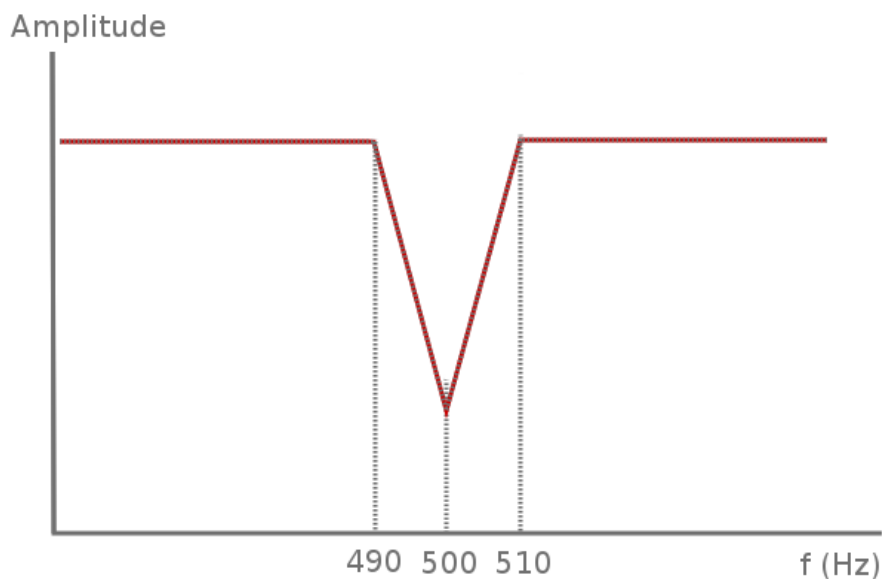


Figure 2 : Forme générale du filtre souhaité

On peut obtenir cette forme en combinant 4 filtres d'ordre 1 : Un filtre passe-bas de fréquence de coupure 490Hz, deux filtres passe-bas inversés de fréquence de coupure 500Hz, et un filtre passe-haut de fréquence 510Hz. Cependant, afin d'obtenir l'atténuation souhaitée, ces filtres doivent être utilisés à des ordres très importants (ordre 20 ou 30). Les tentatives sous Matlab d'obtenir de tels filtres ont malheureusement échoué :

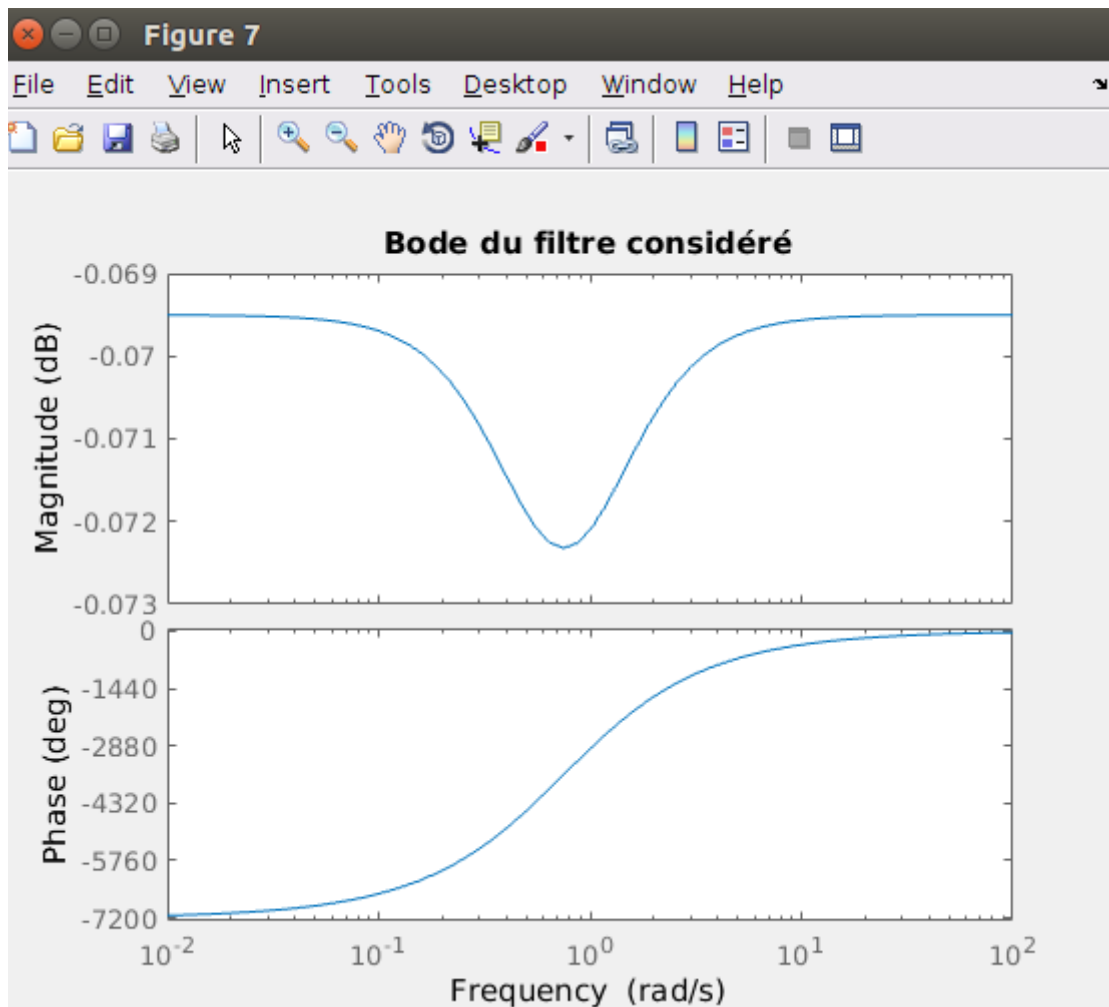


Fig 3 : Caractérisation du filtre 'maison'

Le filtre a bien la forme souhaitée, mais la fréquence est fautive et l'atténuation est mauvaise. Ceci est dû à l'étroitesse du pic à supprimer. Le filtre fonctionne un peu mieux sur des bandes de fréquence plus larges. Il ne sera cependant pas utilisable dans notre cas.

[Script Matlab de la tentative](#)

Au vu de nos connaissances en filtres analogiques, il semble pour le moment impossible de concevoir 'à la main' un filtre plus efficace. Devant l'échec de la deuxième méthode, on se rabat sur la dernière solution : L'utilisation de filtres déjà existants.

2) Filtre Notch

Matlab propose une multitude de filtres ayant fait leurs preuves, et même un outil de conception de filtre.

Dans notre cas, un filtre semble entièrement adapté : Le filtre Notch. Ce filtre permet d'éliminer une bande de fréquence très étroite. Le graphique ci-dessous donne une idée de son fonctionnement :

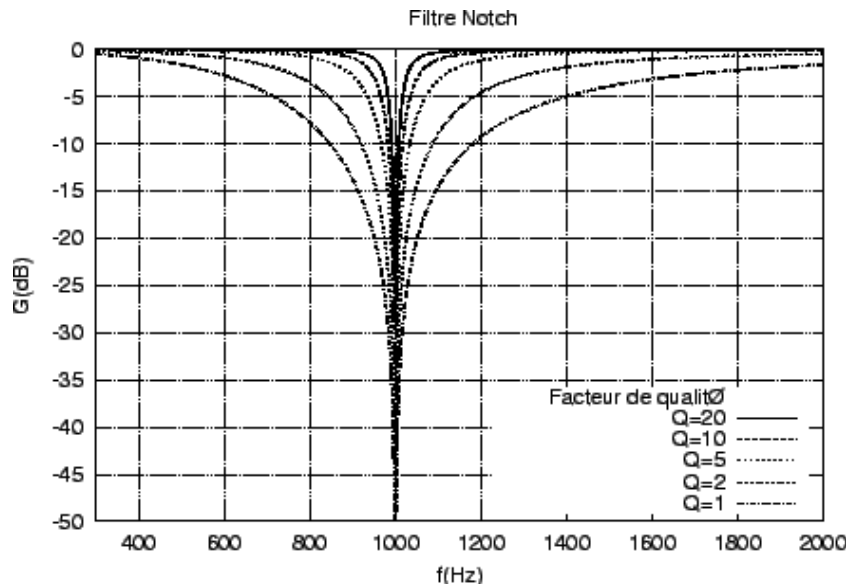


Figure 4 : Principe du filtre notch

Un filtre Notch se dimensionne selon deux paramètres : La fréquence centrale de la bande à couper, et le facteur de qualité. Ce dernier détermine la largeur de la bande coupée à 0dB. Plus le facteur de qualité est élevé, plus cette bande sera étroite. Cependant, cela a tendance à rendre le signal désagréable à l'oreille.

Ce [script Matlab](#) met en place un tel filtre. Le facteur de qualité a été déterminé 'à l'oreille'.

Les étapes du script sont les suivantes :

- **Importation des données du fichier sonore**

- **Détermination des paramètres importants (T_e , F_e , etc.) :**

Ces paramètres sont nécessaires pour les calculs suivants et rendent le script adaptable.

- **Normalisation du signal :**

On ramène toutes les amplitudes entre 0 et 1 pour des raisons de facilité.

- **Tracé du signal d'origine et de sa FFT :**

Cela permet d'avoir une idée du signal à filtrer et offre la possibilité d'adapter le script à d'autres signaux.

- **Calcul des coefficients du filtre Notch (fonction `iirnotch` de Matlab) :**

La fonction `iirnotch` écrit dans deux vecteurs : `num` et `den`. Ces vecteurs contiennent les coefficients du numérateur et du dénominateur de la fonction de transfert en z du filtre.

- Filtrage du signal (fonction filter de Matlab) :

On se retrouve avec un autre vecteur d'amplitude. C'est le signal filtré.

- Tracé de la FFT du signal filtré et écriture du fichier .wav de sortie

Une fois le filtre correctement dimensionné, le [signal de sortie](#) est parfaitement audible. Son spectre est le suivant :

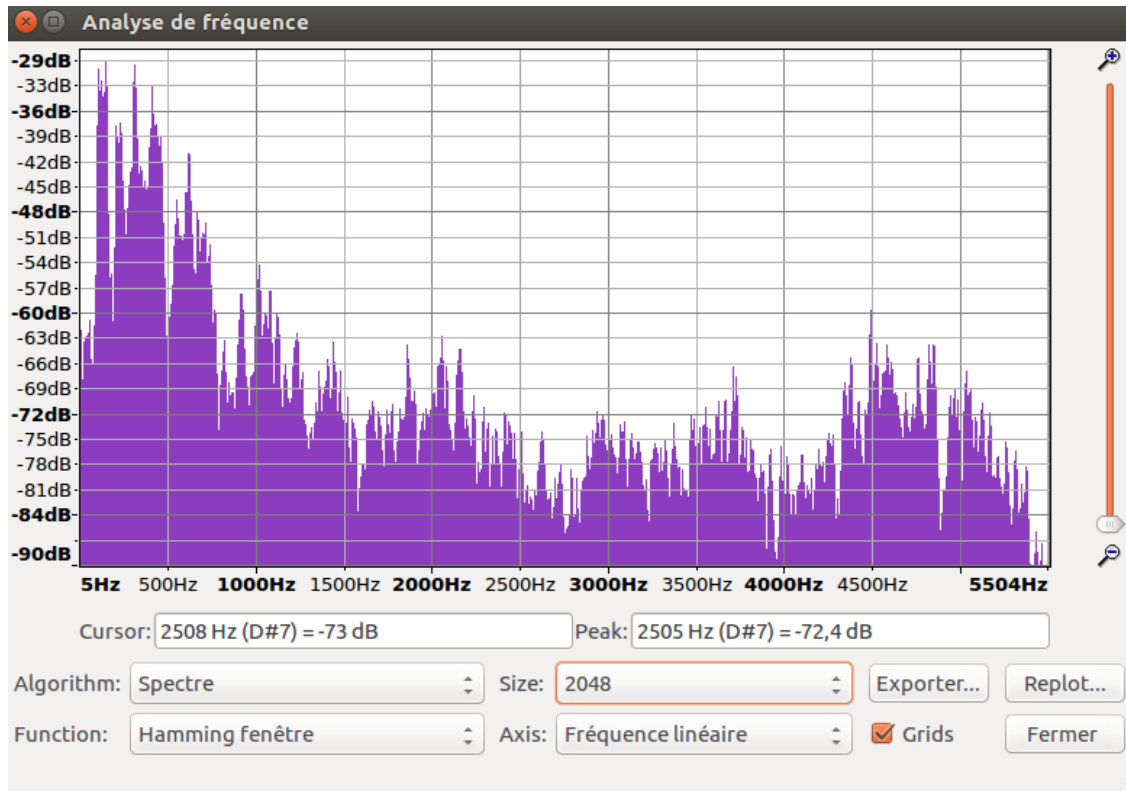


Figure 5 : Spectre du signal filtré

On observe clairement la disparition du pic de fréquence à 500Hz. Les autres fréquences semblent globalement inchangées.

Conclusion

Bien que la plupart des ordinateurs soient assez puissants pour effectuer des transformées de Fourier précises, le filtrage numérique reste la technique la plus employée pour modifier un signal au niveau fréquentiel. Il est bien entendu possible de concevoir ses propres filtres mais les filtres déjà existants sont suffisamment bien pensés pour être plus efficaces que n'importe quel filtre 'maison'. J'ai personnellement été surpris de l'efficacité du filtrage employé. Le signal est en effet passé de 'totalement inaudible' à 'parfaitement audible' en une seule opération de filtrage, ce qui à mes yeux est incroyable.