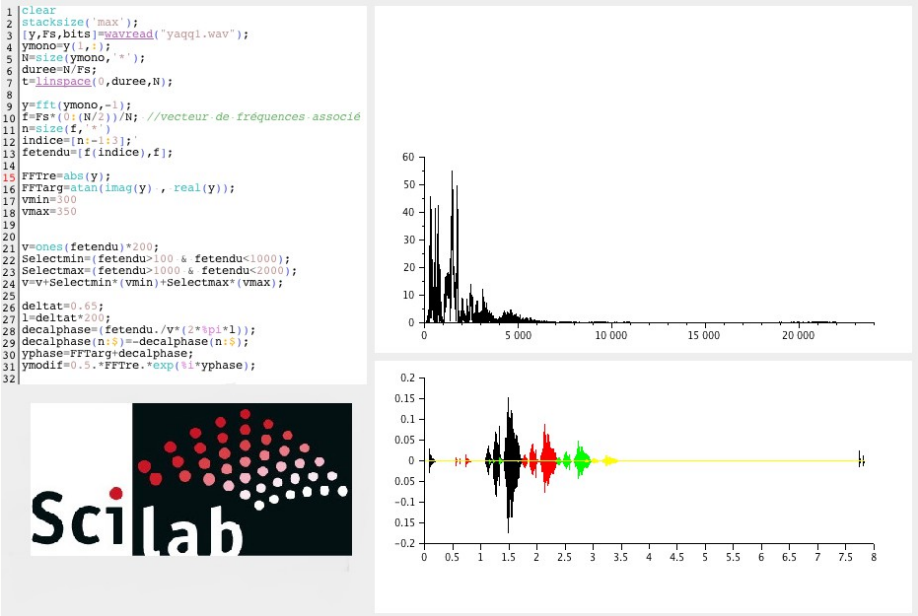
Projet de Physique P6

STPI/P6/2014 – 46

Le traitement numérique de fichiers audios



Etudiants :

Lou AZEVEDO DA SILVA Killian JAUBERT

Morgane LEGER Matthieu MARTINS-BALTAR

Markus NEUPERT

Enseignant-responsable du projet :

Jérôme YON

Date de remise du rapport : **16*/06/2014***

Référence du projet : ***STPI/P6/2014 – 46***

Intitulé du projet : ***Le traitement numérique de fichiers audios***

Type de projet : ***Expérimental / Modélisation***

Objectifs du projet :

Comprendre ce qu'est le son et comment il se propage.

Modéliser une onde sonore en utilisant des outils mathématiques.

Comprendre le principe de filtres fréquentiels (analyse de Fourier) et l'appliquer.

Prendre en main le logiciel Scilab pour traiter les fichiers audios.

Appliquer les outils et filtres au traitement des fichiers audios pour modéliser des effets sur le son ou créer des atmosphères différentes.

Mots-clefs du projet : ***son, onde, Fourier, filtre.***

INSTITUT NATIONAL DES SCIENCES APPLIQUEES DE ROUEN
Département Sciences et Techniques Pour l'Ingénieur
BP 8 – avenue de l'Université - 76801 Saint Etienne du Rouvray - tél : +33(0) 2 32 95 66 21 - fax : +33(0) 2 32 95 66 31

TABLE DES MATIERES

NOTATIONS, ACRONYMES

1. INTRODUCTION

Dans le cadre de projets de P6, nous avons étudié les possibilités de traitement numérique d'un fichier audio.

Le but de ce projet est donc de manipuler des fichiers audio informatiquement. Aujourd'hui, le traitement numérique des sons est de plus en plus utilisés dans les domaines de la musique, la télévision, etc.

L'objectif est d'expérimenter, comprendre et expliquer différents exemples de modifications d'un signal sonore. Il existe différentes manières d'approcher un signal sonore et par conséquent différentes manières de le traiter.

Nous avons choisi d'aborder trois types de traitements numériques ; tout d'abord, la modélisation d'un phénomène naturel, l'écho. Ensuite, nous avons reproduit un outil très utilisé: l'égaliseur. Enfin, nous avons tenté de modifier la hauteur des notes d'un enregistrement. Pour ce faire, nous avons utilisé deux approches, une approche fréquentielle et une autre temporelle.

2. MÉTHODOLOGIE / ORGANISATION DU TRAVAIL

Pendant les deux premières séances, nous nous sommes concertés afin de choisir les différentes directions à prendre pour notre projet. Le sujet étant vaste, il a fallu sélectionner des exemples à traiter.

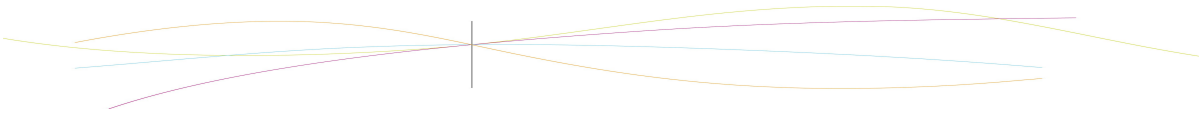
Ensuite, quelques séances ont été dédiées à la recherche bibliographique sur le son et à la découverte du logiciel « Scilab », le logiciel de calcul numérique que nous avons utilisé pour traiter les signaux. En effet, cet environnement nous étant inconnu, il nous fallait connaître ses possibilités, ses limites, et quelques commandes de base.

Afin d'explorer au maximum les trois pistes de traitement que nous avons choisies, nous nous sommes répartis le travail comme suit :

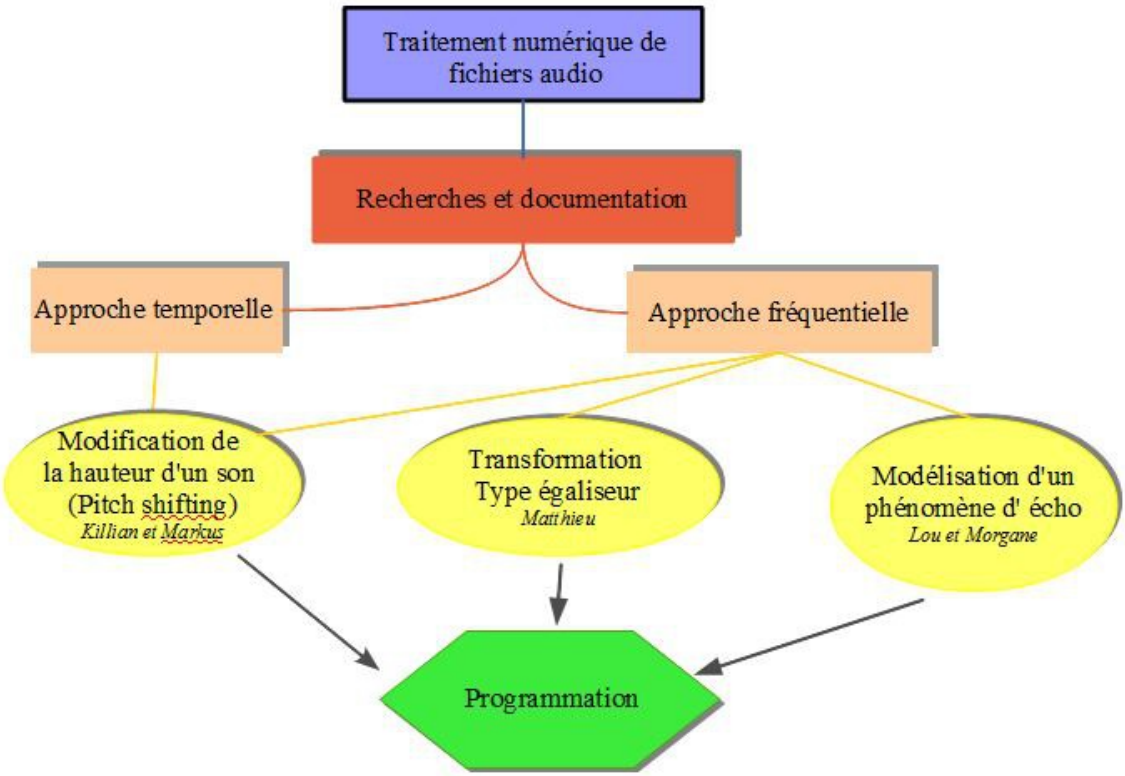
- Morgane et Lou se sont occupées de modéliser le phénomène d'écho,
- Matthieu s'est chargé de réaliser un égaliseur,
- Markus et Killian ont modifié la hauteur d'un son. Ils ont abordé ce travail de deux façons : Killian a utilisé une approche fréquentielle, tandis que Markus s'est intéressé à l'évolution temporelle du signal sonore.

A chaque début de séance, chaque groupe a expliqué aux autres où il en était et ce qu'il souhaitait faire durant la séance.

Une fois les programmes implémentés, nous nous sommes concentrés sur la rédaction du rapport de ce projet.



Organigramme de répartition du travail durant le projet



3. TRAVAIL RÉALISÉ ET RÉSULTATS

3.1. Recherches théoriques

3.1.1. Le son

- ***Définition du son***

Le son est une vibration mécanique d'un fluide, qui se propage sous forme d'ondes longitudinales. Cette propagation est rendue possible grâce à la déformation élastique de ce fluide. Il s'agit là de la tendance d'un matériau à retrouver sa forme d'origine après avoir été déformé par des forces extérieures.

Considérons un milieu fluide compressible (air, eau, etc.). Le son se propage alors sous forme de variation de pression. Cette compression se propage longitudinalement, et les particules du fluide oscillent autour d'une position stable.

Les solides peuvent également transmettre un son, à l'aide de vibrations, ce qui résulte en une contrainte du matériau.

La vitesse de propagation du son est directement influencée par la masse volumique et la constante d'élasticité du milieu. Plus ces facteurs sont grands, et plus la vitesse est petite.

- ***Modélisation***

Pour évaluer la « perception sonore », on transforme la pression acoustique en un signal électrique à l'aide d'un microphone.

On peut alors définir le signal dans l'espace temporel, ou dans l'espace fréquentiel.

Lorsqu'on étudie les variations de la pression de l'air, il s'agit d'une représentation appelée « spectre de modulation d'amplitude ». La variation des fréquences sonores en fonction du temps, est appelé sonagramme. Dans les deux cas, on représente l'histoire de la valeur du signal.

L'espace fréquentiel permet d'analyser le spectre du signal, qui est obtenu grâce à la transformation de Fourier. Le spectre représente les fréquences des différentes sinusoïdes (sons purs), qui reconstitueraient le signal si on les ajoutait. Ce spectre nous sera utile dans chacune de nos expérimentations.

3.1.2. Le traitement numérique du son

- ***Représentation informatique du son :***

Pour une représentation informatique, il faut convertir le signal en valeurs numériques. Ce procédé est appelé « numérisation du son » ou « échantillonnage », car il consiste à relever des petits échantillons de son à des intervalles de temps précis. Cet intervalle de temps est appelé taux d'échantillonnage. Dû au grand nombre d'échantillons par seconde (nécessaires pour restituer un son qui semble continu à l'oreille humaine), on raisonne plutôt sur le nombre d'échantillons par seconde, exprimé en Hertz. Cette donnée est récupérée lors de la lecture du fichier dans nos programmes afin de ne pas altérer le son initial.

Taux d'échantillonnage	Qualité du son
44 100 Hz	Qualité CD
22 000 Hz	Qualité radio
8 000 Hz	Qualité téléphone

Tableau 1 : Rapports entre taux d'échantillonnage et qualité

Une valeur, déterminant la valeur de la pression de l'air, est associée à chaque échantillon. Le son est donc représenté comme une suite de valeurs pour chaque intervalle de temps.

L'ordinateur travaillant avec des bits, il faut fixer le nombre de valeurs que cet échantillon peut prendre :

- 8 bits : 256 valeurs possibles
- 16 bits : 65536 valeurs possibles

Plus il y a de valeurs possibles, plus le format d'encodage est de qualité, est plus il aura besoin de mémoire de stockage.

- ***Traitement numérique :***

L'ordinateur est un outil puissant, donnant accès à de nombreux algorithmes permettant de travailler le signal (rapidement), comme des filtres ou des transformées. De plus, les signaux codés numériquement ne sont pas bruités lors de leur traitement.

Concrètement, le traitement numérique du son a une place importante dans le monde d'aujourd'hui. Il est utilisé pour :

- Stockage de données audio
- Compression
- Transmission (ex. : radio)
- Transformation (ex. : égaliseur, réduction/annulation de bruit, effets…)

Il existe de nombreux logiciels permettant d'appliquer ou de créer des effets sonores, du plus accessible, étant sans doute Audacity, jusqu'aux studios professionnels tels que *FL Studio*. Cependant, pour avoir une influence plus directe sur la manipulation, il est aussi possible de manier des fichiers audio à l'aide de programmes de calcul numériques.Lors de notre projet, nous avons utilisé le logiciel Scilab, un logiciel libre de calcul numérique pour les applications scientifiques. Dans le cadre de notre projet, il nous a permis d'utiliser une large palette d'outils mathématiques appliqués aux signaux sonores.

3.1.3. Les outils mathématiques

- ***La transformée de Fourier***

Pour pouvoir manipuler l'espace des fréquences pour traiter numériquement les sons, nous avons utilisé des outils mathématiques adaptés à cette grandeur. L'outil que nous avons utilisé est la transformée de Fourier.

La transformée de Fourier est une fonction d'analyse qui transforme une fonction intégrable sur l'ensemble des réels ou celui des complexes en une autre fonction, qui met en évidence le spectre des fréquences de cette fonction.

Il existe plusieurs variantes de cette fonction. En physique, on utilise la définition suivante :

$$FFT(f)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} f(t)e^{-i\omega t}dt$$

FFT étant la transformée de Fourier, f la fonction intégrable, t la variable de temps et ω la pulsation.

On remarque que ω=2πf et que par conséquent, la transformée de Fourier dépend directement de la fréquence de la fonction. Comme on intègre par rapport au temps, elle ne dépend plus de celui-ci.

Une propriété importante de la transformée de Fourier est que celle-ci est linéaire. Ainsi, si on somme deux fonctions f et g, alors $FFT(f+g)=FFT(f)+FFT(g)$

Lorsque l'on somme les intégrales pour chacune des fréquences pour lesquelles est définie la fonction f, on obtient le spectre fréquentiel de la fonction.

Une fois ces outils compris et maîtrisés, nous avons toutes les cartes en main pour réaliser nos expériences de traitement du son.

3.2. Expériences et traitements réalisés

Nous allons maintenant développer les différentes démarches qui ont menées à la réalisation de nos traitements de fichiers audio.

3.2.1. Modélisation d'un phénomène réel : l'écho

- ***Principe du phénomène***

Le phénomène d'écho est un phénomène de la vie quotidienne. Vous en avez sûrement déjà fait l'expérience à la montagne ou dans une pièce carrelée par exemple. Un écho, c'est simplement la réflexion d'une onde sonore sur un obstacle dur de surface de taille très supérieure à sa longueur d'onde. Avant de revenir à notre oreille, le son est réfléchi.

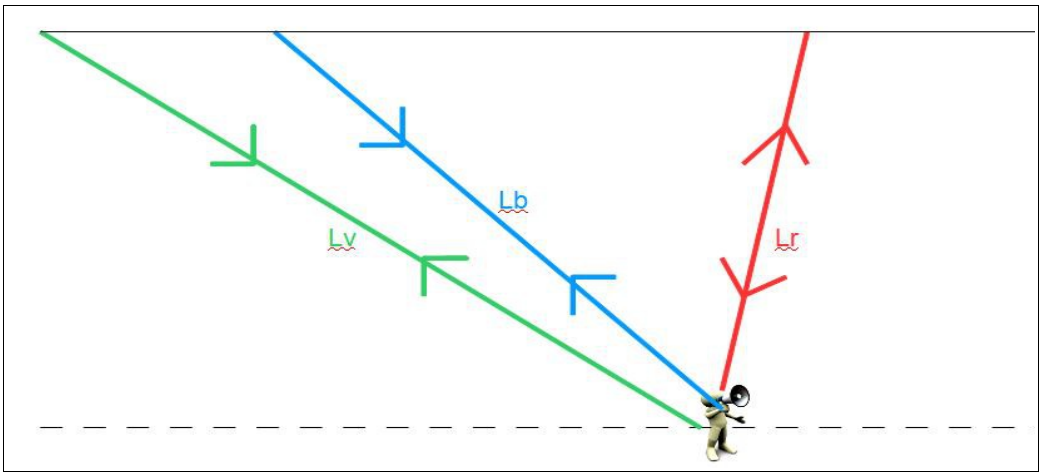


Illustration 1 : Schéma illustrant le phénomène d'écho sur une paroi.

Pourquoi lorsqu'un son est réfléchi sur une surface, il nous parvient avec un retard, et est répété plusieurs fois ? Quand le son est émis à la source, une multitude d'ondes va se propager rectilignement jusqu'à rencontrer l'obstacle. Les ondes se propagent toutes avec une même vitesse, et dans toutes les directions.

Sur le schéma ci-dessus sont représentées les trajectoires de 3 ondes différentes. Lv, Lb et Lr représentent la distance parcourue par chaque onde entre la source et la paroi. On voit que Lv > Lb > Lr, et que l'onde parcourt deux fois chaque longueur avant de revenir à notre oreille. Comme $v = \frac{L}{t} \rightarrow t = \frac{L}{v}$, on voit bien que l'onde « verte » mettra plus de temps à parvenir à notre oreille que l'onde « bleue » et l'onde « rouge ».

Mais ce retard n'est pas le seul phénomène qui se produit lors d'un écho sur un mur. En effet, en rencontrant l'obstacle, l'onde perd également une partie de son énergie dans le matériaux; on appelle ce phénomène l'absorption.

En plus de ce phénomène, les différentes fréquences ne voyagent pas toutes à la même vitesse : c'est l'effet de dispersion de l'onde. Bien que cet effet ne soit pas perceptible

dans l'air, nous avons choisi de le modéliser dans notre expérience. Ces deux raisons expliquent pourquoi le son qui revient aux oreilles paraît étiré et atténué.

En résumé, pour modéliser un écho, il faut créer un décalage temporel du son, puis modéliser la dispersion et l'absorption de l'onde sonore.

- ***Décalage temporel***

Notre objectif dans cette première étape était donc de décaler le signal sonore dans le temps, c'est-à-dire de lui donner un retard. Pour pouvoir percevoir ce retard, nous avons limité le signal sonore entre 0,2 et 0,3 secondes. Le signal est suivi de silence.

En physique, la variation de pression lorsqu'une onde sonore se propage, est notée comme ceci :

$$P(t) = Ae^{i(\omega t - kx)}$$

où A est l'amplitude de l'onde sonore, t la variable de temps et x la variable de longueur.

Un son quelconque (une voix par exemple) est constitué d'une multitude d'ondes exprimées comme celle-ci, qui s'additionnent.

Le but ici est d'exprimer la variation de pression en fonction du temps à une distance x=L de la source sonore, à partir de l'expression du signal de base. On cherche à exprimer ce retard en fonction des fréquences, pour pouvoir appliquer le retard à la transformée de Fourier.

$$P(t) = Ae^{(i(\omega t - kx))}$$

Or on sait que $k = \frac{\omega}{v}$, ainsi $P(t) = Ae^{iw(t-\frac{L}{v})} = Ae^{i\omega t} * e^{-i\frac{\omega L}{v}}$

en remplaçant ω par 2Πf, on obtient :

$$P(t) = Ae^{i\omega t} * e^{-i\frac{2\pi L}{v}f}$$

Dans ce signal, on trouve deux termes : le premier terme, $Ae^{i\omega t}$ est l'expression du signal à la distance x=0, c'est-à-dire l'expression du signal sonore à la source. Il n'a pas de retard.

Lorsqu'on multiplie le deuxième terme au premier, on ajoute une phase supplémentaire au signal, ce qui crée un retard.

Ainsi, il suffit de multiplier l'expression du signal sonore à sa source par $e^{-i\frac{2\pi L}{v}f}$ pour avoir l'expression du signal à x=L, avec le retard associé.

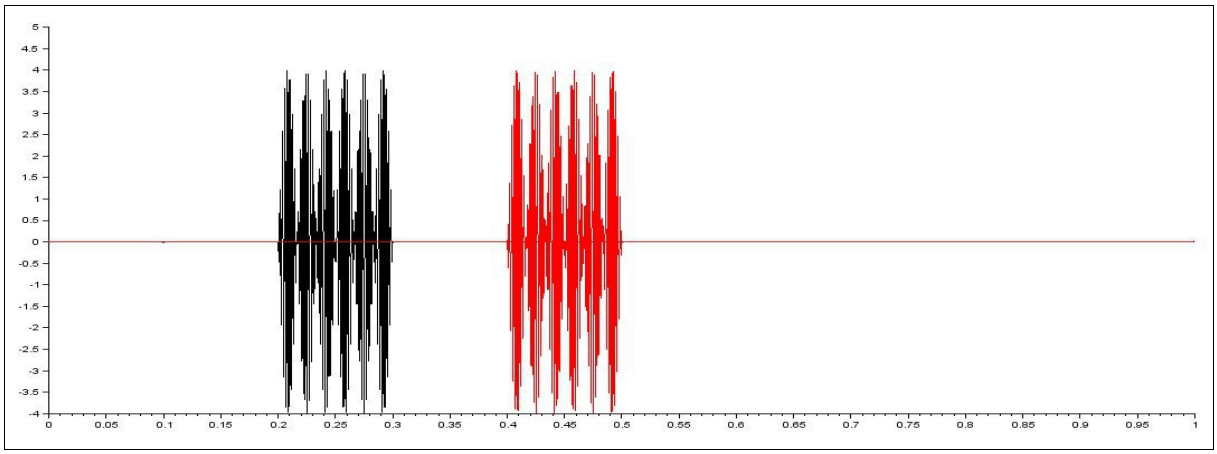


Illustration 2 : En noir, le signal sonore à x=0 et en rouge, le même signal sonore à x=L (avec un retard de 0,2 secondes).

Dans la pratique, pour obtenir ce son décalé, nous sommes passés par la transformée de Fourier de ce signal.

Pour implémenter ce décalage temporel, nous avons distingué d'une part le module du vecteur obtenu par la transformée de Fourier, et d'autre part l'argument de ce vecteur. On obtient l'argument en calculant $Arg(u) = \arctan\left(\frac{\Im(u)}{\Re(u)}\right)$ (Rappelons que la transformée de Fourier renvoie un vecteur qui est un nombre complexe).

Pour décaler le signal sonore, c'est cet argument qui est modifié. Le module, lui, qui correspond à l'amplitude de l'onde sonore, reste inchangée.

Ainsi, pour créer le décalage, après avoir appliqué la fonction Transformée de Fourier au son initial, on ajoute à l'argument de la transformée la phase supplémentaire trouvée précédemment, $\frac{-2\pi L}{v} f$.

Nous avons dû prendre ici en compte le caractère antisymétrique de l'argument de la transformée de Fourier (alors que le module de celle-ci est symétrique en fonction des fréquences). Plus précisément, il a fallu multiplier par -1 les phases supplémentaires associées à la moitié des fréquences (car celles-ci sont construites par symétrie).

Comme nous voulions modéliser l'effet d'écho le plus proche possible de la réalité, nous avons décalé temporellement le son original plusieurs fois, avec des retards temporels différents.

- ***Modélisation de la dispersion de l'onde sonore***

La dispersion d'une onde est l'effet qui correspond à la différence de vitesse de propagation pour chaque fréquence. Nous avons voulu faire dépendre la vitesse de la fréquence.

Pour pouvoir modifier la vitesse des fréquences qui nous intéressaient, il a fallu passer par la transformée de Fourier et sélectionner les fréquences prédominantes du signal sonore.

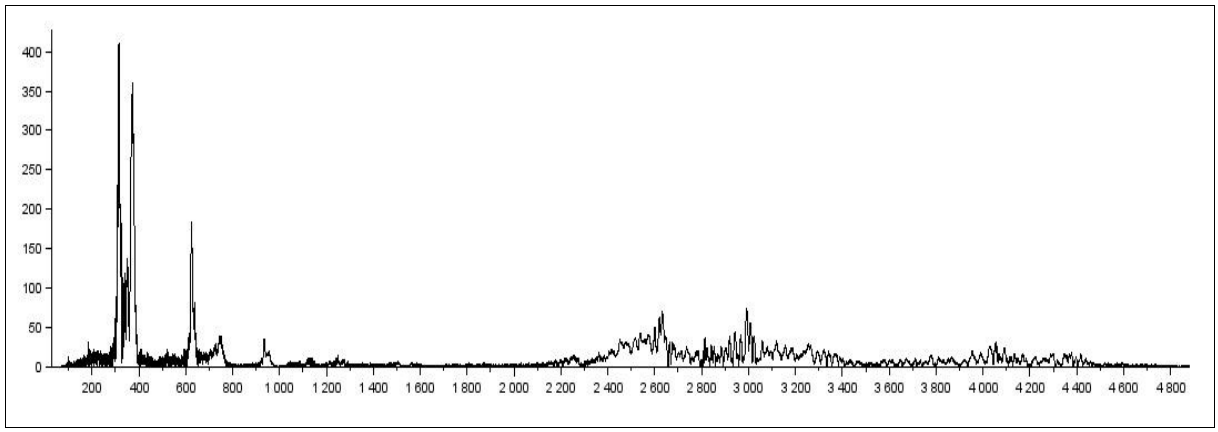


Illustration 3 : Résultat de l'analyse de Fourier d'un signal (en abscisse, la fréquence).

Comme on peut le constater sur l'illustration 3, les fréquences définissant notre son se situent entre 200 et 800 Hz.

Notre première idée a donc été de supprimer les fréquences faibles , entourant la plage de fréquence qui nous intéressait. Nous avons ensuite séparé en deux la plage de données située donc entre 200Hz (fmin) et 800 Hz (fmax) pour l'exemple. Sur l'illustration 4, la fréquence correspondante à la coupure est nommé ftransition. Nous avons alors multiplié la première partie, c'est à dire celle correspondante aux basses fréquences, par une vitesse relativement faible (ici 300m/s) . La seconde partie, correspondante aux hautes fréquences a elle été multipliée par une plus haute vitesse, proche de la vitesse moyenne de propagation d'un son dans l'air, 350m/s.

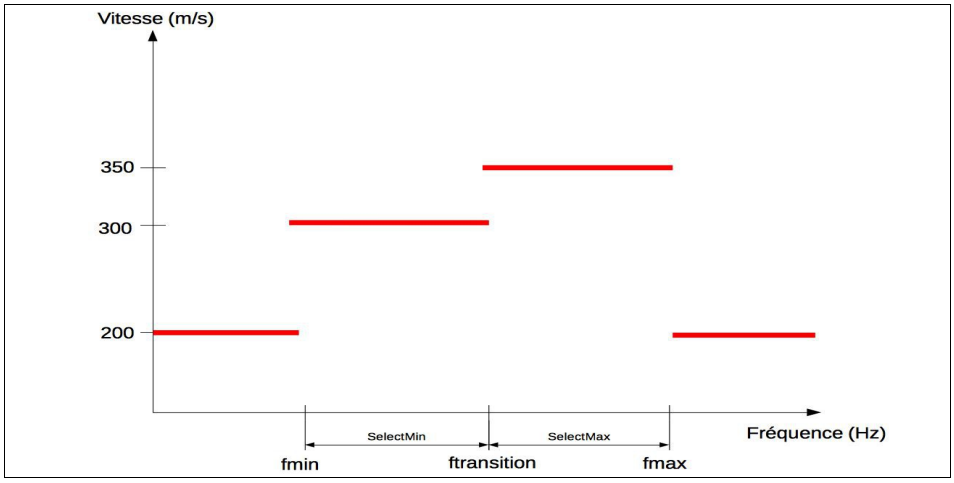


Illustration 4 : Graphique représentant la vitesse de propagation en fonction de la fréquence de l'onde.

Avec cette technique, plusieurs de nos calculs nécessitant une division ne fonctionnaient pas à cause des valeurs nulles présentent dans le vecteur. C'est pourquoi nous nous sommes aperçu que supprimer des fréquences était trop violent. Ces fréquences, bien que peu audibles, se propagent également. Nous avons donc préféré atténuer la propagation de ces fonctions on les multipliant par une faible vitesse, 200m/s ici.

- ***Absorption***

Afin de modéliser l'absorption, nous avons voulu modifier l'amplitude de chaque onde réfléchie. De plus, chaque onde réfléchie, qui vient de nouveau faire un rebond sur la surface, vient à nouveau s'atténuer. D'où la nécessité d'amoindrir l'amplitude un peu plus pour chaque onde sonore revenant à l'auditeur.

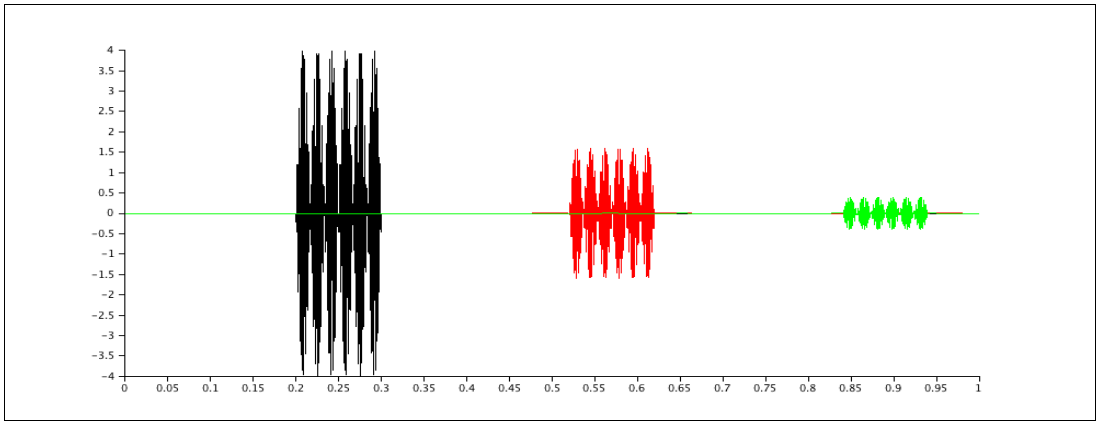


Illustration 5 : Représentation de l'amplitude de 3 signaux sonores en fonction du temps

Afin de modifier l'amplitude de notre onde sonore, il a fallu, encore une fois, faire une transformée de Fourier, puis récupérer la partie réelle de cette transformée. La partie réelle de la transformée correspond à l'amplitude du signal sonore. Il suffit ensuite de multiplier par un terme inférieur à 1 notre amplitude afin de l'atténuer.

Sur l'illustration 4, on peut voir le signal d'origine, sans modification, en noir, d'un son que nous avons généré virtuellement. Le son rouge voit son amplitude diminuer et sa phase modifiée. De même pour le son vert avec des changements d'amplitude et de phase accentués pour donner l'effet réel d'écho.

Nous avons ici principalement traité la phase d'un signal sonore, mais d'autres traitements s'appuient plus sur la modification de l'amplitude d'un signal.

3.2.2. Implémentation d'un égaliseur

Si vous avez déjà manipulé du matériel hi-fi, alors vous savez probablement déjà ce qu'est un égaliseur (equalizer). Il s'agit de la fonction qui permet d'augmenter le volume de certaines fréquences d'onde, par exemple de remonter les fréquences basses. Cela permet de corriger un défaut de matériel, de valoriser l'harmonique principale d'un instrument, de le mettre en avant ou alors de « colorer » le son. Colorer un son sert à favoriser les fréquences qui se propagent le mieux selon le lieux où est placé le dispositif sonore ou bien alors pour apporter un confort personnel sur un style de musique particulier, souvent avec des préréglages comme le réglage « sourire », pour de la musique techno. Le système peut être logiciel ou matériel et se présente généralement sous la forme d'une série de curseurs −appelés aussi *faders*− à monter ou descendre. Les fréquences audibles sont divisées selon une échelle logarithmique et chaque curseur correspond à une portion que l'on peut diminuer ou augmenter.

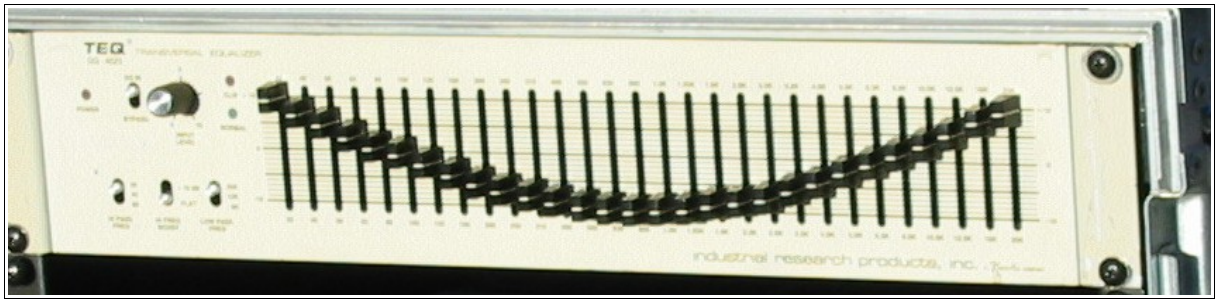


Illustration 6 : Coloration de son de type « sourire » à l'aide d'un égaliseur matériel.

Un égaliseur a pour but d'agir sur les fréquences, il est donc nécessaire, dans une première étape, que le signal subisse une transformée de Fourier pour obtenir un spectre de fréquences sur lequel nous pouvons agir avant de le retranscrire en un signal modifié. Il faut donc pour cela conserver les phases pour replacer temporellement les sons. Le spectre se présentant sous la forme d'une série de « pics », d'harmoniques, représentant la quantité présente de chaque fréquence du spectre, il nous suffit de baisser ou élever les pics compris dans les fourchettes de fréquences que nous voulons modifier.

Pour pouvoir affaiblir et renforcer certaines harmoniques nous avons décidé d'appliquer la méthode suivante : multiplier la partie réelle de la transformée de Fourier par une fonction de transfert en escaliers correspondant aux différents « curseurs ». Chaque palier correspondant à un coefficient multiplicateur pour la fourchette de fréquence qu'il représente. Il existe aussi des égaliseurs paramétriques qui permettent de choisir des points et des vecteurs pour dessiner vectoriellement sa fonction de transfert, mais ce cas de figure ne sera pas abordé.

La seconde étape a donc été de créer cette fonction en tenant compte des coefficients choisis par l'utilisateur de la fonction. Dans un premier temps, pour des raisons de facilité, l'échelle des fréquences n'est divisé que de manière linéaire.

Cela revient donc à diviser l'échelle des fréquences en n parties égales puis de construire une fonction constante par morceaux égale au coefficient choisi pour chaque morceau de l'échelle des fréquences.

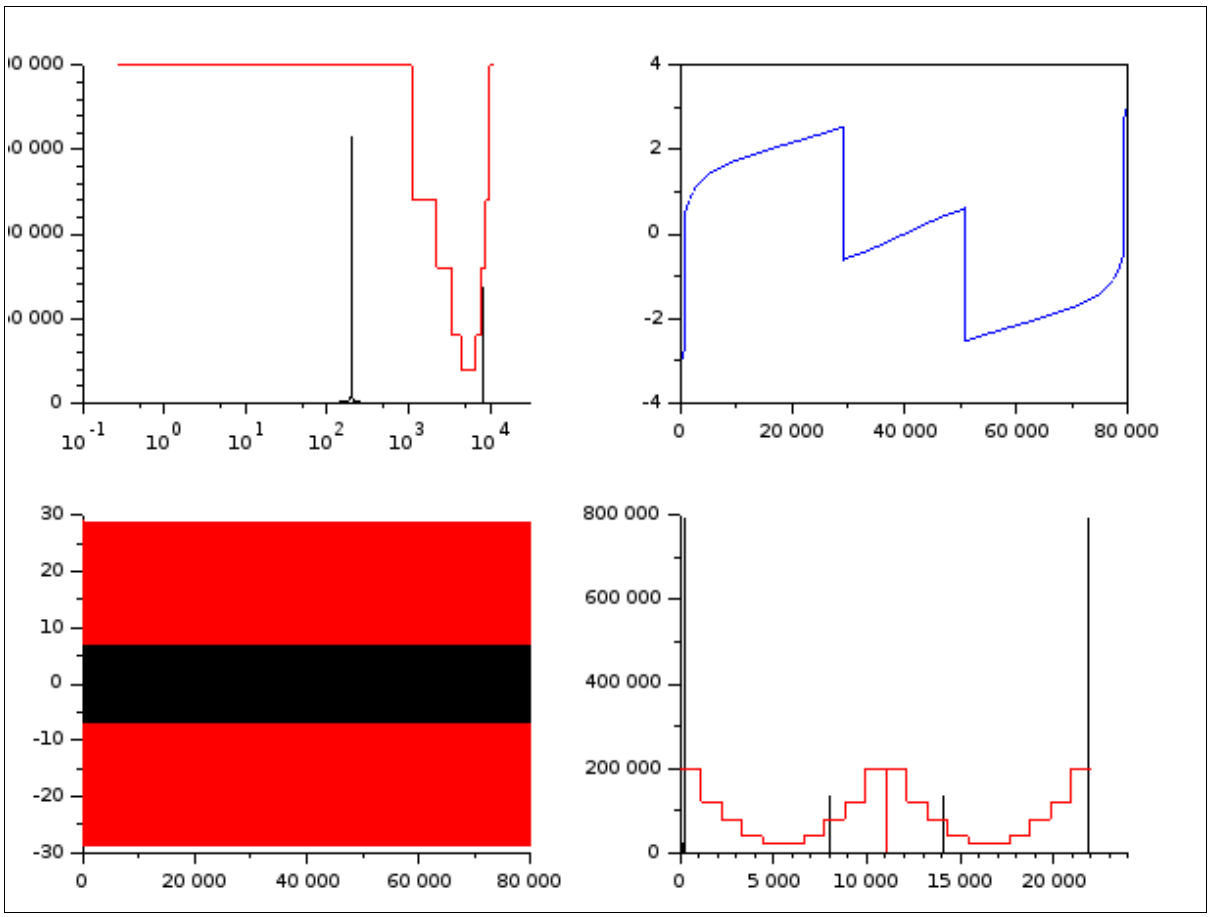
$$\text{Soient } \alpha_i \; i \in [\![0 ; n-1]\!] \text{ les coefficients choisis par l'utilisateur}$$

$$\text{Posons } f = [0 ; \frac{f_e}{2}] \text{ l'échelle des fréquences utilisables dans un fichier audio,}$$

$$f_e \text{ étant la fréquence d'échantillonage}$$

$$\text{Posons ensuite la fonction } t : \begin{matrix} f \to \Re \\ x \in [i(\frac{f_e}{2n}) ; (i+1)(\frac{f_e}{2n})] \to \alpha_i \end{matrix}$$

Une fois cette fonction construite, il ne reste plus qu'à la multiplier au spectre de fourrier.



<u>Illustration 7 :</u> Exemple d'égalisation linéaire d'un son généré.

En haut à gauche : Spectre de Fourir en noir et fonction de transfert en rouge, échelle logarithmique
En haut à droite L'argument de la transformé de Fourier
En bas à gauche : Le signal en fonction du temps, originel en noir et transformé en rouge
En bas à droite : Le spectre du signal transformé en noir, et la fonction de transfert en rouge et leurs symétriques.

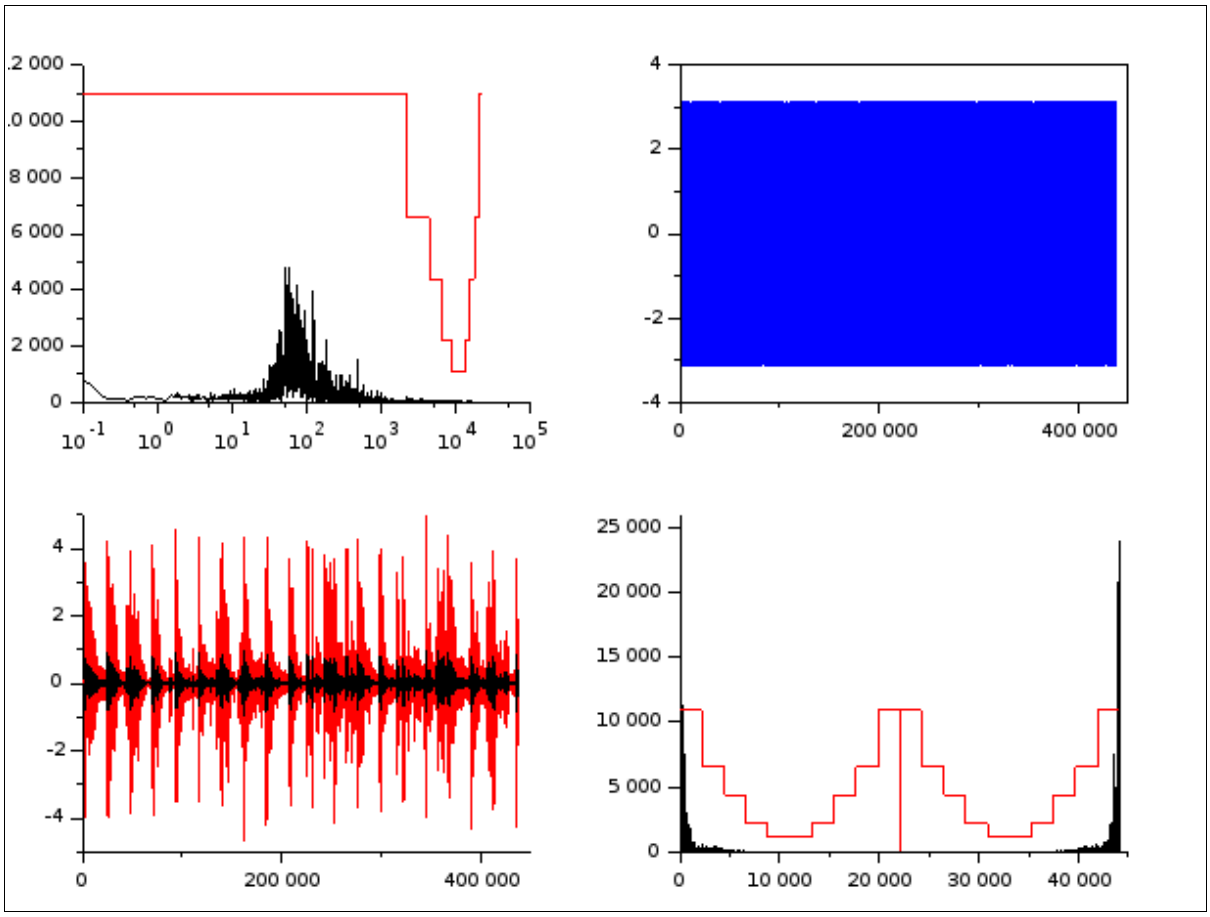


Illustration 8 : Exemple d'égalisation linéaire sur un son réel, très peu efficace.

Pour améliorer l'égaliseur, on peut diviser l'échelle des fréquences de manière plus logarithmique. Nous avons utilisé pour ce faire des fourchettes de fréquences pré-enregistrées. Il existe plusieurs types classiques d'égaliseurs basés sur le nombre de curseurs et les fourchettes de chaque curseur. Nous avons choisi dans notre exemple d'utiliser 10 curseurs et d'utiliser les fourchettes de fréquences suivantes :

16Hz, 32Hz, 63Hz, 125Hz, 250Hz, 500Hz, 1kHz, 2kHz, 4kHz, 8kHz, 16kHz.

Il suffit alors de construire une fonction constante par morceaux, associant un coefficient à chacune de ces fourchettes.

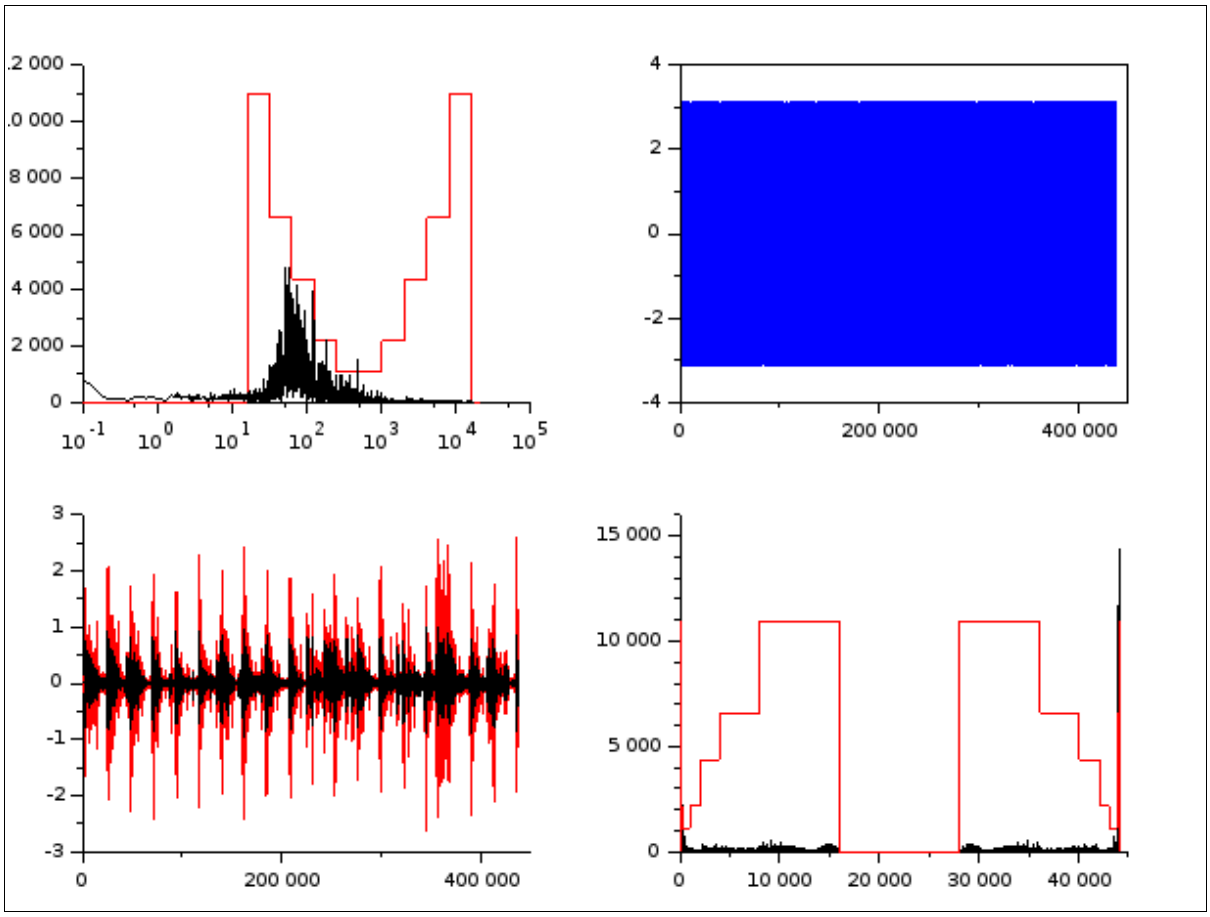


Illustration 9 : Exemple d'égalisation logarithmique sur un son réel.

L'un des principaux problèmes rencontrés dans cette partie du projet, a été le fait que pour certaines grandes fréquences, le spectre débordait du cadre étudié pour être remplacé par son symétrique. Après plusieurs essais, nous nous sommes aperçus qu'en réalité, cela concernait toutes les valeurs de fréquence qui dépassent la moité de la fréquence d'échantillonnage, qui constitue donc la fréquence maximum enregistrable dans le fichier. Dans le cadre de notre projet, nous utilisons des fichiers avec une fréquence d'échantillonnage de 44.1kHz, donc la fréquence maximale est de 22.05kHz.

Après avoir modifié l'amplitude et la phase des ondes sonores, nous pouvons modifier la hauteur de ces sons.

3.2.3. Modification de la hauteur d'un son

- ***Décalage de fréquence***

La hauteur d'une note, appelée aussi le « pitch », est définie par sa fréquence. Par conséquent, pour modifier le pitch il est nécessaire de changer la fréquence d'une note ou d'une musique. Ainsi, nous avons utilisé la transformée de Fourier, afin de pouvoir travailler sur les différentes fréquences. Pour déplacer la hauteur d'une note ou d'une musique, il nous faut donc déplacer son spectre fréquentiel.

En pratique, le spectre fréquentiel de la musique est stocké dans un vecteur contenant les amplitudes de chaque fréquence présentes dans le morceau. Il est important de noter que la transformée est symétrique, c'est-à-dire que la première valeur du vecteur est égale à la dernière et ainsi de suite jusqu'à la valeur du milieu.Tout traitement doit ainsi être fait de part et d'autre du vecteur. Nous avons donc choisi de changer les valeurs de ce vecteur.

La modification que nous avons appliquée, consistait à décaler les valeurs du vecteur. Pour ce faire, nous avons fixé un pas *p* et ainsi donné (dans la première moitié du vecteur) à la *k-ième* valeur du vecteur, l'amplitude de la *k-p* fréquence. Dans la seconde moitié du vecteur, nous avons de ce fait, donné à la *k-ième* valeur, l'amplitude de la *k+p* fréquence. Nous avons évidemment mis a zéro les valeurs des *p* premières fréquences et *p* dernières. En faisant ainsi, nous avons choisi d'augmenter les valeurs des notes, mais nous aurions pu aussi les diminuer.

Nous avons ainsi expérimenté cela sur deux types de signal sonore, un signal périodique simple, composé d'une somme de sinus, que nous avons généré à l'aide de SciLab, et d'une musique disponible dans le commerce.

Son artificiel :

Nous avons choisis de créer un son via SciLab comme étant une somme de sinus de formule $A \times \sin(2\pi \times \omega + \varphi)$, A étant l'amplitude du signal, ω sa fréquence et φ sa phase. Pour essayer de créer un son le plus proche possible de la réalité nous avons décidé que les fréquences des différents signaux composant le son soient des multiples de la fréquence initiale et que leur amplitude soit égale à l'inverse de leur fréquence. C'est ce qui se passe notamment lorsque l'on gratte une corde de guitare par exemple. Ainsi, dans l'exemple suivant, nous avons créé un son composé de 6 signaux, ou 6 harmoniques, avec pour fréquence de base 220Hz, ce qui correspond environ a un *La* sur un piano.

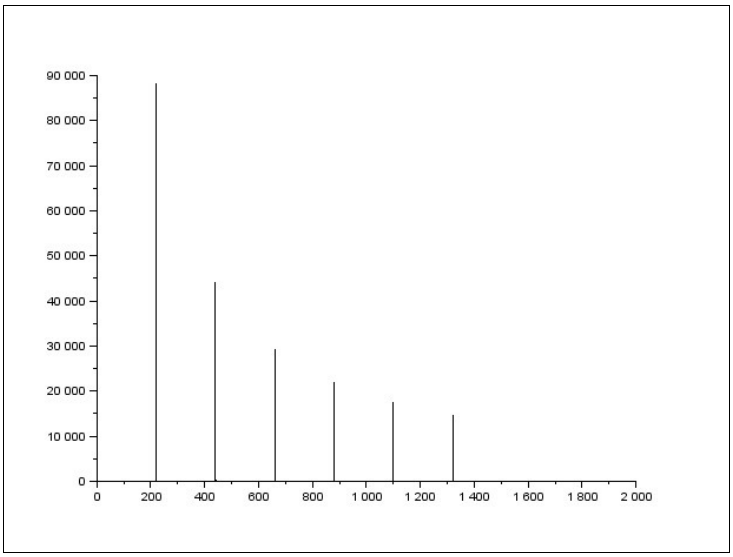


Illustration 10 : Spectre fréquentiel originel.

On retrouve bien dans cette illustration les 6 fréquences des 6 harmoniques dont est composé notre son. Dans l'illustration 11 qui suit, on peut remarquer qu'après avoir appliqué notre traitement, les fréquences ont bien été déplacées.

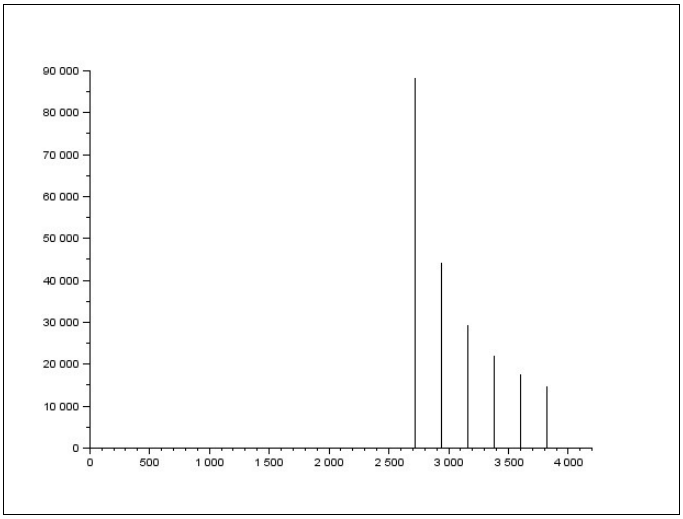
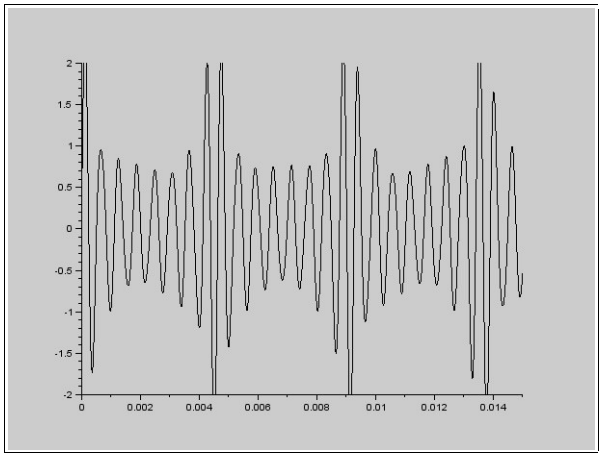
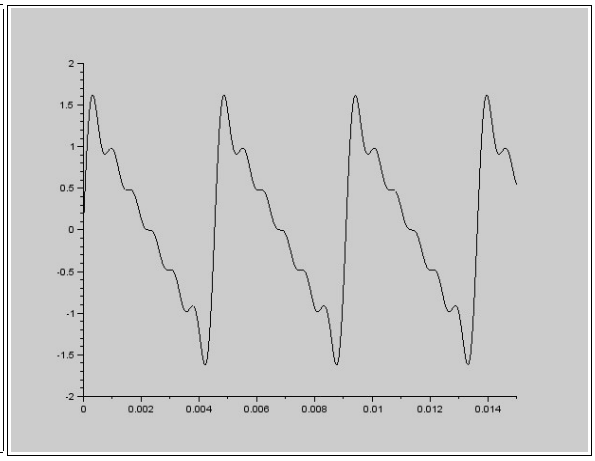


Illustration 11 : Spectre fréquentiel avec un pas de 10 000.

Ainsi, si l'on compare la forme des signaux temporels originel et modifié présentée sur les illustrations 12 et 13 à la page suivante, on remarque bien que le signal a changé et que les fréquences ont très clairement été augmentées. On peut voir un plus grand nombre d'oscillations pour une même durée.



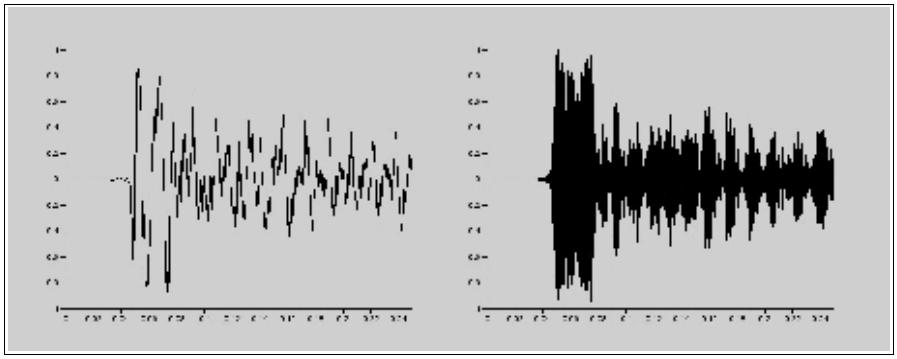
<u>Illustration 12:</u> Son modifié avec un pas de 10 000 sur 1,5ms.



<u>Illustration 13 :</u> Son originel sur 1,5ms.

<u>Musique commerciale :</u>

Notre traitement s'applique également correctement à une musique que l'on trouve dans le commerce. Nous avons obtenu les différents graphiques suivants :



<u>Illustration 14 :</u> Son originel et son modifié avec un pas de 10.000 sur les premières 25ms de la musique.

Sur l'illustration 15, à la page suivante, on remarque bien que l'on a obtenu le même résultat qu'avec un son généré par l'ordinateur : les fréquences ont bien été décalées. D'ailleurs, lorsque l'on écoute la musique modifiée, on entend bien que la hauteur des notes a été augmentée.

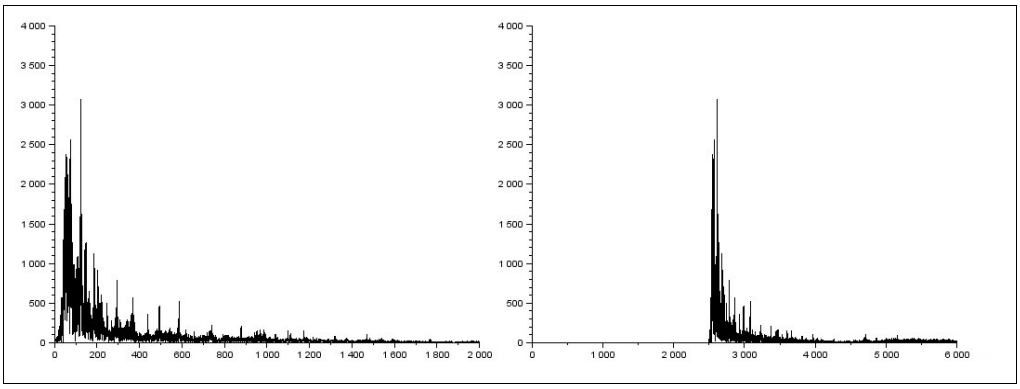


Illustration 15 : Spectre fréquentiel originel et spectre modifié avec un pas de 10.000.

Nous pensions au début que ce décalage était peu précis, c'est-à-dire que la *x-ième* valeur du vecteur ne représentait pas la fréquence *x*. Si l'on fixait par exemple un pas de 200, nous n'augmentions pas les fréquences de 200Hz (On peut notamment voir cela sur les illustrations 11 ou 15). Cependant, après plusieurs expérimentations, nous nous sommes rendus compte que le décalage était en fait tout a fait précis et qu'il répondait à une certaine loi. En effet, en jouant sur la durée du son artificiel, que l'on définissait nous-même, nous avons remarqué que le décalage en fréquence, pour un même pas, était différent selon les durées. Ainsi, nous avons pu constater que le décalage des fréquences répondait à la loi suivante : $Nouvelle\ Fréquence = Ancienne\ Fréquence + \frac{Pas}{Durée}$

Dans nos exemples la durée des deux sons est de 4 secondes. Ainsi, lorsque l'on fixe un pas de 10.000, on augmente les fréquences 2.500. On voit bien cela dans les illustrations 8 et 13.

Nous avons, avec ce traitement, changé uniformément la hauteur d'une musique mais nous aurions également pu modifier certaines fréquences et pas d'autres afin de créer une atmosphère complètement différente. Mais ceci nous semblait plus difficile et plus long a mettre en place.

- ***Étirement temporel***

Une autre méthode pour affecter la hauteur d'un son, est tout simplement de modifier la vitesse de lecture du fichier audio. A la manière d'un DJ qui manipule ses platines et qui ralentit ou accélère cette vitesse, nous pouvons appliquer un effet similaire à nos fichiers, que l'on appellera « étirement temporel ».

Ce procédé se base sur l'informatisation du son ; en effet, c'est en intervenant sur le couple échantillons/fréquence d'échantillonnage que nous pouvons atteindre notre objectif. La fréquence d'échantillonnage correspond au nombre d'échantillons lus par unité de temps (ici, la seconde). En modifiant le nombre d'échantillons, on affecte directement la durée du fichier audio, ainsi que son spectre de fréquences.

Pour mettre cet effet en place à l'aide de Scilab, nous nous sommes servis du vecteur contenant les informations du fichier audio. Il suffit alors de retrancher ou bien d'ajouter des valeurs dans ce vecteur.

Exemple d'application :

Nous avons multiplié la longueur d'un fichier audio par deux, en créant un nouveau vecteur, qui faisait deux fois la taille du vecteur du fichier traité. Nous l'avons alors rempli en attribuant deux nouvelles cases à chaque valeur du vecteur d'origine. Autrement dit, chaque donnée a été dupliquée.

Le résultat était un son deux fois plus long qu'initialement, et bien plus grave.

Nous avons également pu créer un fichier deux fois plus court, en supprimant une valeur sur deux dans le vecteur original.

Il faut noter que cette méthode endommage la qualité du fichier, puisqu'il s'agit d'une suppression ou d'une répétition de données.

4. CONCLUSIONS ET PERSPECTIVES

4.1. Conclusion sur le travail réalisé

Le but de ce projet était de traiter numériquement un son. Au cours de nos expériences, nous nous sommes rendu compte de l'étendu des possibilités offertes par le traitement numérique du son. En effet, il est possible d'obtenir une multitude de sons différents à partie d'un seul en changeant ses fréquences, sa vitesse de propagation son amplitude, sa phase. Cela permet de modéliser des phénomènes naturels aussi bien que des outils utilisés dans des domaines tels que la musique, le cinéma (effets sonores), la reconnaissance vocale ou encore la synthèse vocale.

Pour mener à bien nos expériences il nous a fallu établir un base de connaissances dans les domaines de la physique, des mathématiques et surtout de l'informatique. C'est dans ce dernier domaine que nous avons rencontré le plus de difficultés,en effet nous ne connaissions pas toutes les capacités du logiciel.

Par ailleurs, nous avons mis quelques temps avant de commencer nos expériences concrètement(3- 4 semaines). En effet, l'énoncé du sujet était assez vaste et de ce fait il nous a fallu du temps pour trouver la direction dans laquelle nous souhaitions aller.

4.2. Conclusions sur l'apport personnel de cet E.C Projet

Le principal apport personnel est bien évidemment le travail de groupe. C'est une chose a laquelle nous seront forcément confronté dans notre travail d'ingénieur. Il était donc important pour nous de participer à un tel projet. De plus, nous devions travailler avec une grande autonomie. Il fallait donc bien s'organiser individuellement et collectivement afin de mener a bien ce projet. Les différents membres du groupe ont tenu tout de même a exprimer leurs ressenties personnels.

Lou Azevedo Da Silva :

Ce projet a été tout d'abord pour moi l'occasion d'acquérir des connaissances sur le phénomène d'onde sonore. J'ai toujours été intéressée par les effets que l'on pouvait appliquer au son, particulièrement dans le domaine de la musique. Ayant fait un stage dans un studio d'enregistrement, j'ai pu découvrir l'ampleur et la diversité des effets possibles, mais leur fonctionnement m'était jusqu'alors inconnu. Grâce à nos expériences, j'ai pu découvrir comment certains de ces filtres fonctionnaient et nous avons pu les recréer par nous même.

Concernant le travail de groupe, celui-ci n'a pas été des plus évidents à gérer. En effet, nous voulions pouvoir traiter différents aspects du traitement d'onde sonore, c'est pourquoi nous nous sommes divisés. Ceci nous a en effet permis de créer plusieurs effets, mais rend la communication plus importante et difficile. Finalement, après un temps d'adaptation, nous avons tout de même réussis à trouver une réelle cohésion de groupe et à nous intéresser à chaque expérience. Je me suis rendue compte, grâce à ce projet, qu'avec un peu d'organisation, il est possible de travailler rapidement et surtout que le groupe permet de résoudre des problèmes qu'il aurait été presque impossible de résoudre seuls.

Markus Neupert :

Le son est un phénomène physique du quotidien ; cependant, il couvre une multitude de notions du domaine des sciences, qui nous restent inconnus tant qu'on ne s'y intéresse pas de plus près. Ce projet était l'occasion parfaite de découvrir les secrets du son, mais aussi de le manipuler ! En tant que passionné de musique et d'instruments, il était captivant de créer et de modifier des sons à l'aide d'un clavier et d'une souris. J'ai pu me familiariser avec les lois qui régissent le monde sonore, et entrevoir toutes les possibilités qu'il nous offre.

Hormis l'initiation au traitement numérique d'un son, j'ai également eu la possibilité d'élargir mes expériences de travail en groupe ; mener un projet à bien sur une période de plusieurs mois, requiert une organisation réfléchie et efficace, en accord avec les objectifs à atteindre. Là où une problématique précise impose une progression et une distribution hiérarchique du travail, notre sujet nous a autorisé à explorer la thématique du son selon plusieurs axes, nous permettant de mettre en évidence diverses propriétés du son, ainsi que d'établir des parallèles entre elles.

En somme, ce projet a été un enrichissement personnel au niveau de nouvelles connaissances, et de méthodes de travail. Pour finir, une problématique intéressante et une bonne entente au sein de notre groupe ont rendues cette expérience agréable et originale.

Morgane Léger :

J'ai choisi ce projet afin de découvrir l'importance du son, la diversité de cette notion, ainsi que la multitude de traitements qu'on peut lui appliquer, et surtout comment. Ce projet m'a beaucoup intéressé car j'en sais maintenant beaucoup plus sur les effets sonores, et les traitements de signaux. Le sujet était d'autant plus intéressant qu'il liait les domaines de la physique, des mathématiques et de l'informatique. Il a donc fallu être polyvalent pour atteindre les objectifs des traitements que l'ont voulait effectuer.

Le fait de travailler en groupe durant ce projet a été enrichissant, même si cela n'a pas été toujours facile à gérer. Nous avons dû gérer l'organisation du groupe mais aussi la communication au sein de celui-ci, ce qui n'était pas toujours simple car nous ne nous connaissions pas. Je pense en conclusion que ce projet de physique est un bon aperçu du type de travail que nous allons réaliser dans le futur.

Killian Jaubert :

J'ai trouvé ce projet particulièrement intéressant. En effet, en tant qu'amateur de musique, j'étais assez curieux de voir comment l'on pouvait traiter des sons par ordinateur, et les effets directs de ces traitements. De plus, ce projet m'a permis d'accroître mes connaissances sur le son en général, que je ne connaissais finalement peu. J'ai aussi pu découvrir un nouvel outil de programmation qui, je pense, me sera utile dans le futur.

D'autre part, cette expérience m'a permis de travailler en groupe et avec cela de découvrir un peu plus ce qu'est la réalisation d'un projet. Nous avions des objectifs à se fixer et nous devions donc bien nous organiser pour les atteindre. Je pense tout de même qu'une bonne ambiance régnait dans notre groupe ce qui nous a permis d'avancer assez facilement dans nos travaux.

Matthieu Martins-Baltar :

J'ai trouvé ce projet très intéressant. En effet, il nous a permis de découvrir de manière concrète certaines spécificité du son et de découvrir la transformé de Fourier qui est un outil classique de l'ingénieur dont on se servira lors de nos prochaines années d'études. Cela m'a également permis d'utiliser un nouvel outil de calul mathématiques en plus de matlab utilisé en M8 et sagemath en MAO.

D'autres part le travail en groupe m'a permis d'améliorer mes capacités d'organisation et de communication avec le reste du groupe. Ce n'est pas le premier travail en groupe au sein de l'INSA mais c'est à chaque fois une nouvelle expérience et des contraites différentes à faire face. Par exemple, venant de classes différentes, les gestions d'emploi du temps étaient particulièrement compliqués. Le fait d'avoir des projets en communicaton et en MAO dans les même périodes, s'est également révélé être un challenge à surmonter.

5. BIBLIOGRAPHIE

Recherches théoriques :

Sites Internet :

http://www.lumanmagnum.net/physics/resample.html (valide à la date du 21/04/2014)

http://www.guitarpitchshifter.com/matlab.html (valide à la date du 07/05/2014)

http://fr.wikipedia.org/wiki/Son_(physique) (valide à la date du 13/02/2014)
http://fr.wikipedia.org/wiki/Transformation_de_Fourier (valide à la date du 13/02/2014)
http://fr.wikipedia.org/wiki/Domaine_fréquentiel (valide à la date du 06/03/2014)

6. ANNEXES

6.1. Codes Scilab des programmes réalisés

6.1.1. Modélisation de l'écho

```
clear
stacksize('max');
[y,Fs,bits]=wavread("echoL.wav"); //ouvre le fichier son
ymono=y(1,:);
N=size(ymono,'*');
duree=N/Fs;  // défini la durée du signal
t=linspace(0,duree,N); // vecteur temps associé au fichier

y=fft(ymono,-1);
f=Fs*(0:(N/2))/N; //vecteur de fréquences associé
n=size(f,'*')
indice=[n:-1:3];
fetendu=[f(indice),f];

FFTre=abs(y); // partie réelle de la transformée de Fourier
FFTarg=atan(imag(y) , real(y)); // partie imaginaire
vmin=250
vmax=350


v=ones(fetendu)*200; // on attribut la même vitesse à chaque fréquence
Selectmin=(fetendu>200 & fetendu<400);
Selectmax=(fetendu>600 & fetendu<800);
v=v+Selectmin*(vmin)+Selectmax*(vmax); // on crée notre fonction par palier (v=v(f))

deltat=0.55; // retard du premier écho
l=deltat*200;
decalphase=(fetendu./v*(2*%pi*l)); // phase supplémentaire
decalphase(n:$)=-decalphase(n:$); // on reconstitue la phase en antisymétrie par rapport aux fréquences
yphase=FFTarg+decalphase; // on ajoute la phase supplémentaire à la FFT
ymodif=0.5.*FFTre.*exp(%i*yphase); // on reconstitue la FFT
```

```
deltat=1.1; // même opération avec un nouveau retard
l=deltat*200;
decalphase=(fetendu./v*(2*%pi*l));
decalphase(n:$)=-decalphase(n:$);
yphase=FFTarg+decalphase;
ymodif2=0.3.*FFTre.*exp(%i*yphase);

deltat=1.65; // même opération avec un nouveau retard
l=deltat*200;
decalphase=(fetendu./v*(2*%pi*l));
decalphase(n:$)=-decalphase(n:$);
yphase=FFTarg+decalphase;
ymodif3=0.1.*FFTre.*exp(%i*yphase);

clf
subplot(2,1,1)
plot2d(f,abs(y(1:n)))
subplot(2,1,2)
y2modif=fft(ymodif,1); // transformée de Fourier inverse du signal
y2modif2=fft(ymodif2,1);
y2modif3=fft(ymodif3,1);


echo=(ymono+y2modif+y2modif2+y2modif3) // on somme les signaux
plot2d(t',[ymono',y2modif',y2modif2',y2modif3'],[1,5,3,7])
savewave("echoL-modif.wav",echo,Fs , bits); // enregistrement en fichier audio
```

6.1.2. L'égaliseur

```
clear
stacksize("max")
cd '~/Documents'
function y=fgauss(X,centre,et)
    y=1/sqrt(2*%pi)/et*exp(-0.5*((X-centre)/et).^2)
endfunction


function y=Composinus(X,omega1,A1,omega2,A2)
    y=A1*sin(omega1*X)+A2*sin(omega2*X);
endfunction

function son=creerUnSon(X,amp,Freq,phase)
    son=amp*sin(2*%pi*Freq*X+phase)
endfunction

function y=fonc_transfert_egaliseur(x, e)
    l = length(e) //nombre de fois que l'on divise le graphique
    n = length(x)/length(e) //largeur d'une "barre" du graphique
    
    y=zeros(length(x));
    
    for i = 0:l-1
        y = y + ((X>i*n)& (X<=(i+1)*n))*e(i+1)
    end
    
endfunction

function y=fonc_transfert_eg_log(x, e)
    freqs=[16 32 63 125 250 500 1000 2000 4000 8000 16000]
    
    l = length(freqs) //nombre de fois que l'on divise le graphique
    n = length(x)/length(freqs) //largeur d'une "barre" du graphique
    
    y=zeros(length(x));
    
    for i = 1:l-1
        y = y + ((x>freqs(i))& (x<=freqs(i+1)))*e(i)
    end
    
endfunction

////Avec un son généré
////duree=10;
//nb_points=80000//100000
//freq_echantil=22050;
//duree=nb_points/freq_echantil
////freq_echantil=nb_points/duree
//
//t = linspace(0,duree,nb_points-1)
////t=1:1/freq_echantil:duree;
//N=size(t,'*'); //nombre d'échantillons
//y2=creerUnSon(t,5,200,%pi)+creerUnSon(t,2,8000,%pi)//+creerUnSon(t,5,2000,%pi)+creerUnSon(t,5,500,%pi);
//coef = N/2
////Fin Avec un son généré

//Avec un fichier extérieur
[y2,freq_echantil,bits]=wavread("Get_lucky_10s.wav");
```

```
y2=y2(1,:);
nb_points=size(y2,'*');
duree=nb_points/freq_echantil;
t=linspace(0,duree,nb_points);
N=size(t,'*'); //nombre d'échantillons
coef=N/200 //Serst juste à rendre la fonction de transfert visible sur le graphique
//Fin avec un fichier extérieur

y=fft(y2);
f=freq_echantil*(1:N)/N//vecteur de fréquences associé
n=size(f,'*')

X=[1:N];

clf
FFTre=abs(y);
FFTarg=atan(imag(y) , real(y));

//Transfert = fonc_transfert_egaliseur(t(1:n/2), [5 3 2 1 0.5 0.5 1 2 3 5]);
Transfert = fonc_transfert_eg_log(f, [5 3 2 1 0.5 0.5 1 2 3 5]);

//On affiche le spectre du signal avec la fonction de transfert
subplot(2,2,1)
plot2d(f(1:n/2),Transfert(1:n/2)*coef,5,logflag="ln") //*coef pour que ce soit visible
plot2d(f(1:n/2), FFTre(1:n/2),logflag="ln")

//On affiche l'argument de la transformé de fourrier
subplot(2,2,2)
plot(FFTarg)

//symétrie de la fonction de transfert
Transfert_droite = flipdim(Transfert,2,1)
Transfert = [Transfert(1:N/2) Transfert_droite(N/2:N-1+(modulo(N,2)))]
//-1+(n modulo 2) car si N est impaire, Transfert pourrat être de longueur N-1 au lieu de N

//On applique la transformation
ymodif=FFTre.*Transfert.*exp(%i*FFTarg);
y2modif=fft(ymodif,1);

//On affiche le signal original et le signal modifié.
subplot(2,2,3)
plot2d(X',[y2modif',y2',],[5,1])

//On affiche le spectre transformé et la fonction de transfert avec leurs s
subplot(2,2,4)
//plot2d(f, y)
plot2d(f,abs(ymodif))
plot2d(f, Transfert*coef,5)

////Avec un son généré
//wavwrite(y2modif, freq_echantil, 'sonmodifie.wav')
//wavwrite(y2, freq_echantil,'sondebase')
////Fin Avec un son généré

//Avec un fichier extérieur
savewave("Get_lucky_10s-modif.wav",y2modif,freq_echantil , bits);
//Fin Avec un fichier extérieur
```

6.1.3. Le décalage de fréquence et l'étirement temporel

- **Avec la musique commerciale**

```
//·Chargement·du·fichier·audio·original
[son,ftaux_echantilonage,bits]=wavread("C:\XXX\Musique_commerciale.wav")

//·Implémentations·nécessaires·pour·créer·notre·base·de·temps

N=size(son,'*');·····//Taille·du·vecteur·contenant·le·son
duree=N/taux_échantillonage;·····//Calcul·de·la·durée·du·son
t=linspace(0,duree,N);·····//Creation·de·la·base·de·temps

//·Traitement

fourrier_son=fft(son,-1)·····//Transformée·de·Fourier·sur·le·son
fourrier_modifié=fourrier_son·····//Copie·du·vecteur·pour·le·modifier
L=length(fourrier_son)·····//Taille·du·vecteur·de·la·transformée

Pas = XY·····//Pas·de·décalage·modulable

for i=Pas+1:((L/2)+(Pas-1))·····//Décalage·sur·la·première·moitié·du·
    fourrier_modifié(:,i)=fourrier_son(:,i-pas)·····//vecteur·symétrique
end

for i=1:(Pas)·····//Mise·a·zéros·des·valeurs·qui·ont
    fourrier_modifié(:,i)=0·····//été·décalées
end

for i = ((L/2)+ pas):(L-pas)·····//Décalage·sur·la·deuxième·moitié·du
    fourrier_modifié(:,i)=fourrier_son(:,i+Pas)·····//vecteur·symétrique
end

for i=(L-(Pas-1)): L·····//Mise·a·zéros·des·valeurs·qui·ont·
    fourrier_modifié(:,i)=0·····//été·décalées
end

son_modidifié=fft(fourrier_modifié,1)·····//Transformée·de·Fourrier·inverse·pou
·····//recomposer·la·musique

//·Création·du·fichier·wav.·afin·de·pouvoir·écouter·le·son·modifié

wavwrite(son_modifié,taux_échantillonage,'Son_modifié.wav')
```

- ***Avec le son artificiel***

```
// Implémentation de la fonction pour créer une sinusoïde

function son=creerUnSon(X,amp,Freq,phase)
    son=amp*sin(2*%pi*Freq*X+phase)
endfunction

// Implémentations nécessaires pour créer notre base de temps

duree= XX;                                  //Durée modulable
taux_échantillonage= XY;                    //Taux d'échantillonnage modulable
                                            //(44100Hz pour une meilleure qualité)
t = 0:1/sample_rate:duree;                  //Creation de la base de temps

// Creation du son artificiel

freq_base= XX                               //Fréquence de base de la sinosoïde modulable
son=creerUnSon(t,1,freq_base,0)             //Création premier sinus
for i=2:6                                   //Création de la somme de sinus
    son = son +creerUnSon(t,1/i,i*freq_base,0)
end

// Le traitement se déroule de la même façon

// Création du fichier wav. afin de pouvoir écouter le son modifié

wavwrite(son,taux_échantillonage,'Son_originel.wav')        //Son originel
wavwrite(son_modifié,taux_échantillonage,'Son_modifié.wav') //Son modifié
```