



28 mai 2025

Sara EL BARI - Justine BRISSART

Situation d'apprentissage riche : AUDIO



IMT Atlantique

Bretagne-Pays de la Loire
École Mines-Télécom

1 Introduction générale

L'objectif de cette situation d'apprentissage riche est de mettre en œuvre et d'étudier plusieurs techniques de traitement de signaux audio en utilisant l'outil MATLAB. Ces techniques couvrent différentes approches de la synthèse sonore telles que la synthèse additive et la synthèse soustractive mais aussi des techniques d'application d'effets audio numériques comme l'effet de réverbération et l'effet de retard. Au cours de ce TP, nous cherchons à analyser, observer et parfois recréer des sons en utilisant différents outils théoriques du traitement du signal tels que les Transformées de Fourier et leurs variantes, les filtres ou la convolution.

2 Présentation du problème

Comme expliqué en introduction, nous traitons différents aspects des fichiers audio au cours de ce TP. En premier lieu, nous nous intéressons à la synthèse additive qui consiste à synthétiser un son musical harmonique en additionnant des ondes sinusoïdales. Il s'agit donc de traiter les harmoniques du signal et de manipuler les paramètres de l'enveloppe ADSR (Attack, Decay, Sustain, Release) qu'on lui applique. Cette enveloppe permet de rendre le son plus réaliste. Concernant la synthèse soustractive, on part cette fois d'un signal riche en harmoniques qu'on filtre pour en définir le timbre, c'est-à-dire pour contrôler la bande de fréquences conservées. Ces deux types de synthèses sont comparés pour bien saisir les caractéristiques des sons produits.

La seconde partie porte sur les effets audio numériques appliqués. On modélise tout d'abord un effet de réverbération grâce à la convolution d'un signal avec une réponse impulsionnelle. Ensuite, on crée un délai grâce à un système de rétroaction récursive (filtre IIR) qui simule les réflexions du son. Enfin on ajoute un filtre dans la boucle de retard pour mieux simuler l'absorption des hautes fréquences.

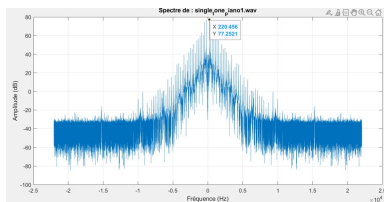
3 Synthèse additive

3.1 Analyse d'un son harmonique

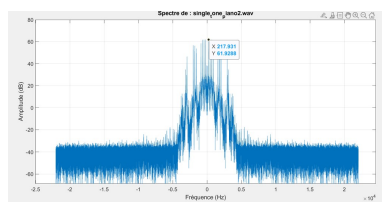
3.1.1 Analyse fréquentielle de sons instrumentaux

On dispose de plusieurs sons d'instruments sur une note pour certains. Pour chacun de ces signaux, on applique la Transformée de Fourier Discrète grâce à la fonction "fft". Ensuite, le spectre est recentré autour de zéro grâce à la fonction "fftshift" (symétrie entre $-f_e/2$ et $+f_e/2$). De plus, on s'assure d'afficher l'amplitude en décibels pour faciliter la lecture des harmoniques. On obtient ainsi un graphique pour chaque spectre avec la fréquence en Hertz et l'amplitude en décibels. On peut alors déterminer la fréquence fondamentale f_1 de chaque instrument

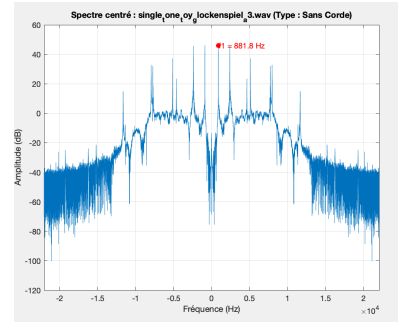
3. Synthèse additive



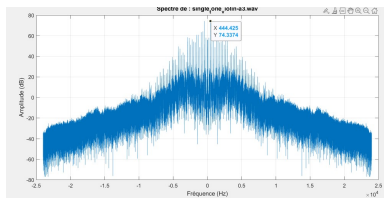
(a) piano1.wav



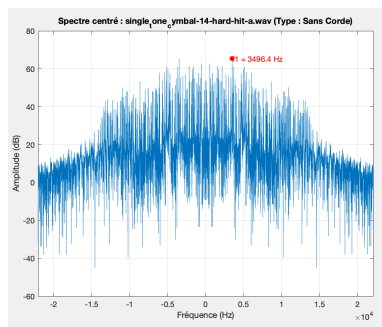
(b) piano2.wav



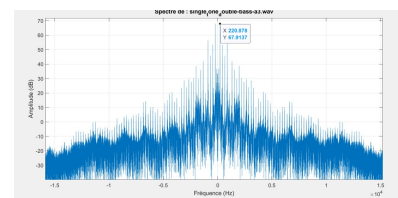
(c) glockenspiel-a3.wav



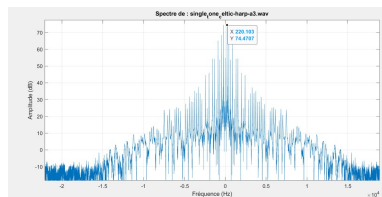
(d) violin-a3.wav



(e) cymbal-14-hard-hit-a.wav



(f) double-bass-a3.wav



(g) celtic-harp-a3.wav

FIGURE 1 – Spectres centrés des différents signaux audio analysés.

Grâce aux spectres obtenus, on peut déterminer la fréquence fondamentale f_1 de chaque instrument. Elle correspond à la plus basse fréquence dans le signal audio.

Pour les sons traités, on a :

- $f_1(\text{cymbale}) = 3496.28 \text{ Hz}$
- $f_1(\text{glockenspiel}) = 881.69 \text{ Hz}$
- $f_1(\text{violin}) = 444.42 \text{ Hz}$
- $f_1(\text{glockenspiel}) = 881.69 \text{ Hz}$
- $f_1(\text{piano1}) = 220.45 \text{ Hz}$
- $f_1(\text{piano2}) = 217.93 \text{ Hz}$
- $f_1(\text{double basse}) = 220.87 \text{ Hz}$
- $f_1(\text{celtic harp}) = 220.10 \text{ Hz}$

3.1.2 Étude de l'inharmonicité : comparaison des sons de piano

Pour cette question on évalue l'inharmonicité des deux pianos grâce à la formule :

$$\xi = 1200 \times \left(\log_2(\hat{f}_n) - \log_2(n \cdot f_1) \right)$$

- \hat{f}_n est la fréquence mesurée de l'harmonique n ;
- f_1 est la fréquence fondamentale du piano (220,45 Hz pour le piano1 et 217,93 Hz pour le piano2) ;
- n est le numéro de l'harmonique.

On peut tout d'abord comparer les spectres centrés des pianos 1 et 2 grâce à leur représentation :

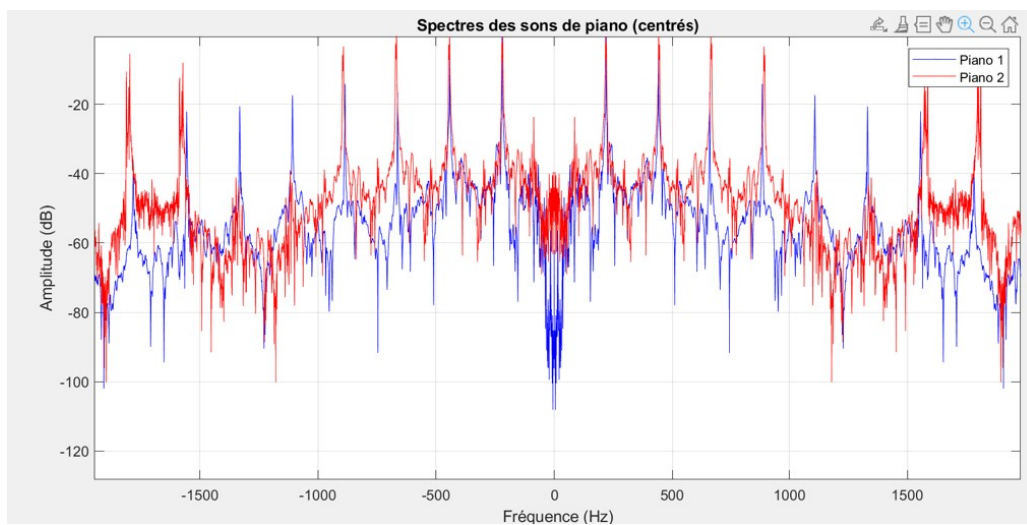


FIGURE 2 – Spectres centrés des sons piano1.wav (bleu) et piano2.wav (rouge) — zoom sur les premières composantes fréquentielles.

A vue d'oeil, on ne distingue pas de différence notable en terme d'harmonicité entre ces deux signaux. Cependant, il semble que le piano 2 a des pics moins nets ou plus irréguliers.

Les calculs des inharmonicités pour les 7 premières harmoniques (faits à la main) nous donnent les résultats suivants :

TABLE 1 – Inharmonicité du son piano1.wav

Harmonique	Fréquence théorique (Hz)	Fréquence mesurée (Hz)	Inharmonicité (cents)
2	440.91	442.04	4.43
3	661.37	663.22	4.85
4	881.82	885.54	7.28
5	1102.30	1108.30	9.49
6	1322.70	1331.50	11.38
7	1543.20	1556.40	14.71

TABLE 2 – Inharmonicité du son `piano2.wav`

Harmonique	Fréquence théorique (Hz)	Fréquence mesurée (Hz)	Inharmonicité (cents)
2	435.86	442.78	27.27
3	653.79	667.42	35.72
4	871.72	892.49	40.76
5	1089.70	1572.00	634.51
6	1307.60	1797.10	550.52
7	1525.50	2022.20	487.93

Ainsi, on remarque que l'inharmonicité reste toujours très faible avec des valeurs inférieures à 15 cents pour le piano 1 tandis que pour le piano 2 si pour les 3 premières harmoniques, l'inharmonicité est inférieure à 40 cents, les valeurs s'envolent et sont de l'ordre de la centaine pour les 3 harmoniques suivantes.

Ainsi le piano 1 est plus harmonique que le piano 2. Le piano 2 est considéré complètement inharmonique.

3.1.3 Synthèse additive à partir des harmoniques du `piano1.wav`

L'objectif est de reconstruire un son synthétique à partir du spectre du son du piano 1 qui présente une structure globalement harmonique.

Le principe de la synthèse additive repose sur la somme de sinusoïdes définies par :

$$s(t) = \sum_{n=1}^8 A_n \cdot \sin(2\pi f_n t)$$

où f_n sont les fréquences mesurées (en Hz) et A_n les amplitudes correspondantes, extraites dans le domaine fréquentiel. Le signal synthétisé est calculé avec durée de deux secondes et une fréquence d'échantillonnage de 44.1 kHz. On le normalise aussi pour éviter la saturation lors de l'écoute.

La figure suivante est un zoom sur les 50 premières millisecondes du signal synthétisé :

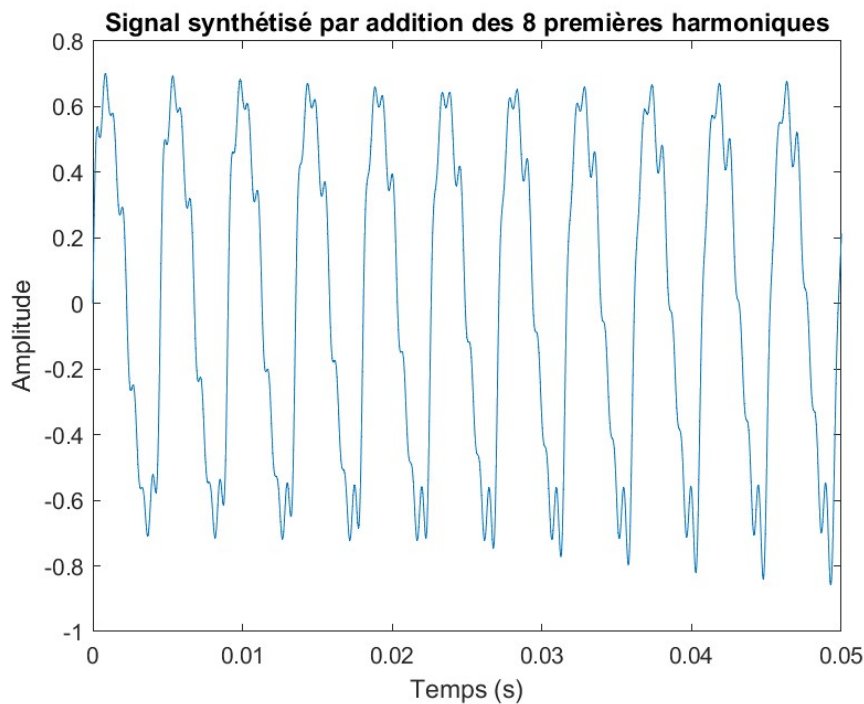


FIGURE 3 – Zoom temporel sur le signal synthétisé par addition des 8 premières harmoniques extraites du `piano1.wav`.

La forme du signal obtenu semble parfaitement périodique. De plus, on observe des irrégularités dans les oscillations qui sont dues à l'ajout des harmoniques supérieures. Cependant, le signal synthétisé n'est pas complètement fidèle à celui de piano 1 : les amplitudes sont constantes dans le son synthétisé, ce qui n'est pas le cas dans le son réel.

3.1.4 Application d'une enveloppe ADSR

Ici, on applique une enveloppe ADSR (*Attack*, *Decay*, *Sustain*, *Release*) pour mieux modéliser les variations de l'intensité sonore du piano avec le temps.

On décide de lui donner les paramètres suivants :

- **Attack** : montée linéaire de l'amplitude de 0 à 1 sur 100 ms,
- **Decay** : décroissance vers un niveau de maintien (0.6) sur 100 ms,
- **Sustain** : maintien de l'amplitude à 60% pendant 1.3 s,
- **Release** : décroissance finale de l'amplitude jusqu'à 0 sur 400 ms.

Les paramètres ont été choisis afin d'imiter le comportement physique de l'instrument. Cette enveloppe est générée comme un vecteur de même taille que le signal, puis appliquée point à point via une multiplication.

La figure ci-dessous présente le signal obtenu après application de l'enveloppe, avec un zoom sur les premières 600 ms.

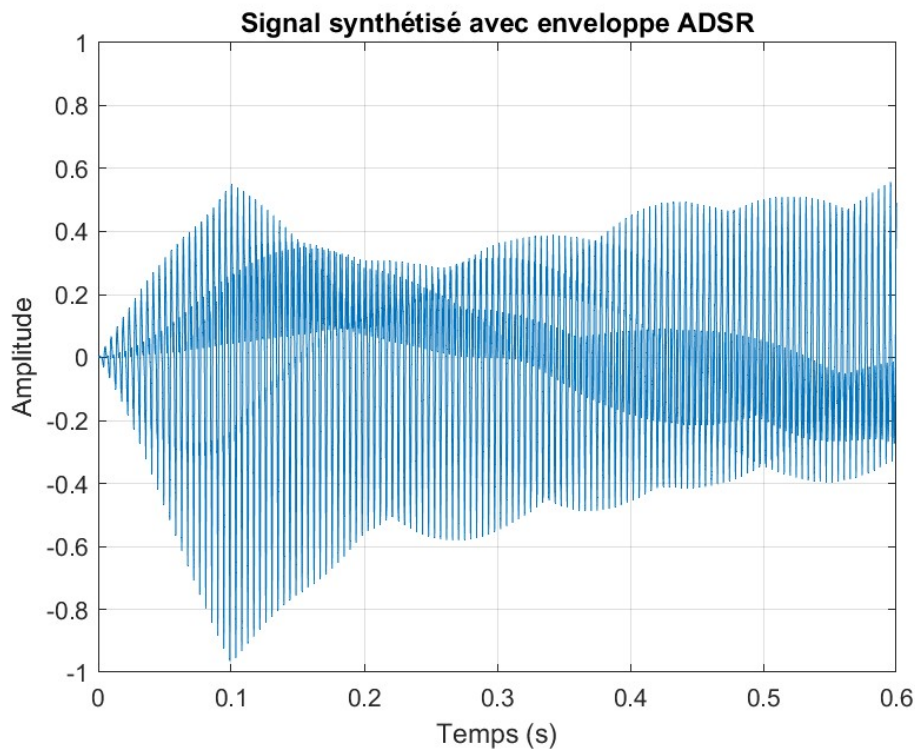


FIGURE 4 – Zoom temporel sur le signal synthétisé avec enveloppe ADSR appliquée.

On observe bien sur la figure que l'amplitude du signal évolue dans le temps. Au début (vers 0s), le son monte progressivement en volume : c'est la phase d'attaque. Ensuite, le signal diminue un peu (décay), puis reste à un niveau plus stable (sustain). Enfin, à la fin de la figure, le signal diminue doucement : c'est la phase de relâchement (release). Le code sur git ajoute cet enveloppe au son, et donc le son paraît plus naturel à l'écoute, car il suit une dynamique proche de celle d'une vraie note de piano.

3.1.5 Synthèse par transformée de Fourier inverse (IFFT)

On génère par Transformée de Fourier Discrète Inverse (IFFT) un signal temporel à partir d'un spectre "artificiel" contenant les huit principales harmoniques du son `piano1.wav`. On a mis les amplitudes mesurées dans un spectre, puis on a ajouté les mêmes composantes de façon symétrique pour que le signal obtenu soit bien réel après la transformation inverse.

La figure 5 montre que le spectre obtenu après reconstruction (IFFT puis FFT) présente des pics aux mêmes fréquences et avec des amplitudes similaires à celles du spectre initial. Cela confirme bien la réversibilité de la transformée de Fourier discrète et la validité de cette méthode de synthèse.

Cependant, le son obtenu reste statique, car aucune enveloppe temporelle (type ADSR) n'a été appliquée.

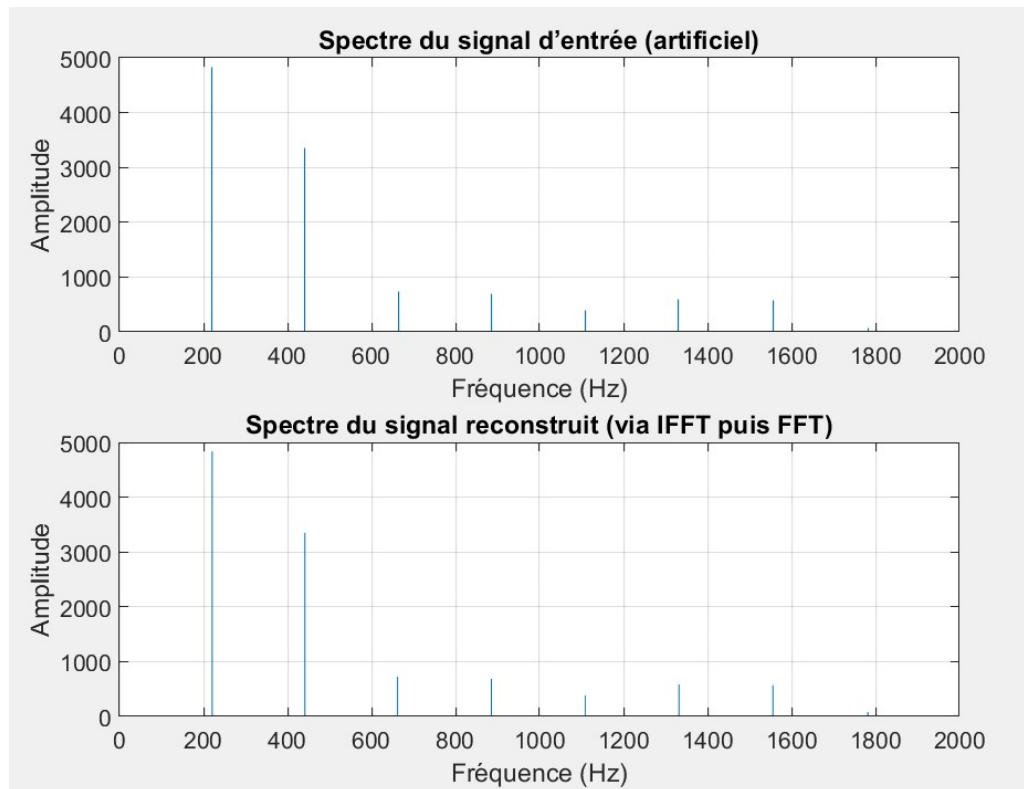


FIGURE 5 – Spectres du signal d'entrée (haut) et du signal reconstruit par IFFT (bas).

4 Synthèse soustractive

4.0.1 Question 2.1

La synthèse soustractive repose sur la génération d'un signal riche en harmoniques, que l'on filtre ensuite pour modéliser son spectre. Nous commençons par analyser deux signaux : le signal carré et le signal dent de scie.

Pour l'analyse théorique du signal carré centré, sa série de Fourier est :

$$x(t) = \begin{cases} 1, & 0 < t < \frac{T}{2} \\ -1, & -\frac{T}{2} < t < 0 \end{cases} \quad \text{et périodique sur } R$$

Le signal est **impair et périodique**, donc sa série de Fourier ne contient que des **sinus** :

$$x(t) = \sum_{n=1}^{\infty} b_n \sin\left(\frac{2\pi n t}{T}\right)$$

D'où le spectre du signal carré :

$$x(t) = \frac{4}{\pi} \sum_{n=1,3,5,\dots}^{\infty} \frac{1}{n} \sin(2\pi n f_1 t)$$

où $f_1 = \frac{1}{T}$ est la fréquence fondamentale. Le spectre contient donc uniquement des **harmoniques impaires** avec une amplitude qui décroît comme $\frac{1}{n}$.

Le signal en dent de scie centré (symétrique en ± 1) est défini par :

$$x(t) = \frac{2t}{T}, \quad -\frac{T}{2} < t < \frac{T}{2}$$

Ce signal est aussi impair, donc sa série de Fourier est :

$$x(t) = \frac{2}{\pi} \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n} \sin(2\pi n f_1 t)$$

Le spectre de la dent de scie contient donc **toutes les harmoniques** (pairs et impairs), avec une décroissance $\frac{1}{n}$ et une **alternance de signe**.

Méthode de vérification numérique

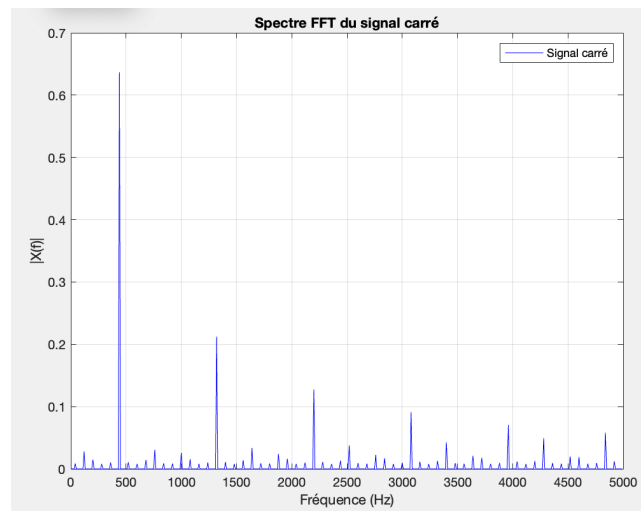


FIGURE 6 – Spectres du signal carré

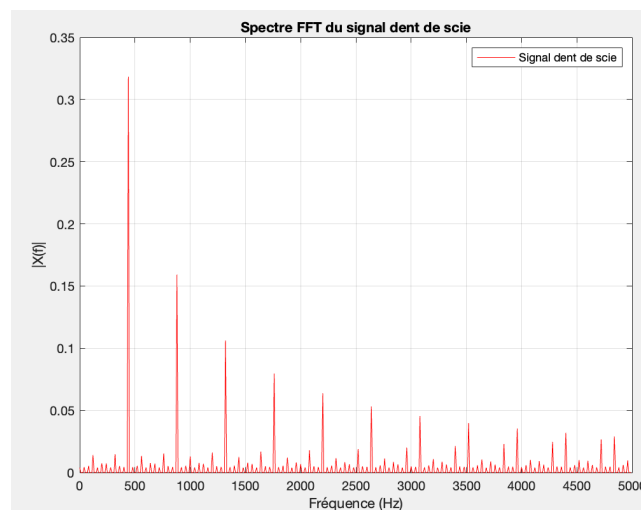


FIGURE 7 – Spectres du signal en dent de scie

Pour valider ces résultats analytiques, on a implémenter les signaux dans MATLAB et visualiser leur spectre avec la transformée de Fourier discrète (FFT). On retrouve les caractéristiques théoriques donc le modèle est bien validé.

4.0.1.1 Limites de cette méthode numérique : Cependant, la méthode numérique présente certaines limites. Par exemple, la FFT suppose que le signal est périodique, ce qui peut créer des discontinuités aux extrémités et entraîner un phénomène de fuite spectrale

En plus de ça, si la fréquence d'échantillonnage est insuffisante, on observe un repliement spectral.

Ce test numérique permet toutefois de confirmer efficacement la structure harmonique de ces deux signaux.

4.0.2 Question 2.2

On a le filtre passe-bas défini par :

$$y(k) = \frac{1}{2} (x(k) + x(k-1))$$

Sa réponse impulsionnelle est donc :

$$h(k) = \frac{1}{2}\delta(k) + \frac{1}{2}\delta(k-1)$$

La transformée de Fourier discrète de $h(k)$ donne la fonction de transfert du filtre $H(f)$:

$$H(f) = \sum_k h(k)e^{-j2\pi f k} = \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot e^{-j2\pi f} = \frac{1}{2}(1 + e^{-j2\pi f})$$

On peut alors calculer le module :

$$|H(f)| = \left| \frac{1}{2}(1 + e^{-j2\pi f}) \right| = |\cos(\pi f)|$$

Donc on déduit que le spectre de sortie est donc :

$$|Y(f)| = |X(f)| \cdot |H(f)| = |X(f)| \cdot |\cos(\pi f)|$$

On observe donc que le filtre réalise un filtrage passe-bas, car :

- pour $f \approx 0$, on a $|H(f)| \approx 1$ (pas d'atténuation) ;
- pour $f = \frac{1}{2}$, on a $|H(f)| = 0$ (annulation complète).

Les 2 signaux, simulé et théorique, sont confondus. On peut donc valider la simulation.

4.0.3 Question 2.3

Dans cette question, nous avons appliqué la même enveloppe ADSR que celle utilisée dans la synthèse additive, sur un signal périodique riche en harmoniques qu'on a filtré avec un filtrage passe-bas, tel qu'un signal carré ou un signal en dent de scie.

On garde la même enveloppe, on teste plusieurs signaux de base en synthèse soustractive (carré, dent de scie et triangulaire) et on le fait aussi pour la synthèse additive.

Après écoute des sons générés, on conclut :

Le signal dent de scie, qui est très riche en harmoniques, donne un son plus brillant, même après filtrage tandis que le signal carré, moins riche, offre un son plus doux.

La synthèse additive, construite en sommant 8 harmoniques contrôlés, produit un son plus propre, plus prévisible, mais moins vivant, cette approche manque du côté de la richesse et chaleur du son.

4.0.4 Question 2.4

Pour améliorer la qualité du son en synthèse soustractive, nous avons exploré trois paramètres du filtre :

(i) fréquence de coupure f_c ; (ii) ordre du filtre N ; (iii) nature {RIF, RII}.

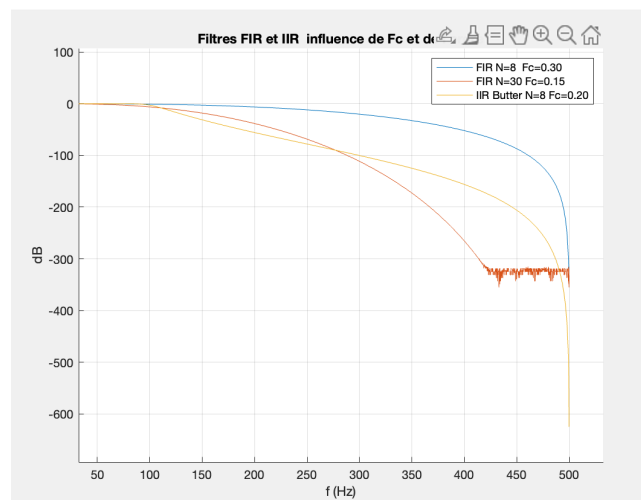


FIGURE 8 – Influence de f_c et de l'ordre et nature du filtre

Le filtre $y(k) = \frac{1}{2}(x(k) + x(k-1))$ est très simple, mais peu sélectif. Un filtre d'ordre supérieur atténue beaucoup mieux les harmoniques non souhaités, ce qui permet d'obtenir un son plus propre, plus proche d'un vrai instrument.

4.0.4.1 Conclusion Si on augmente l'ordre du filtre, la courbe devient plus raide donc une atténuation efficace des harmoniques aiguës tout en conservant un timbre organique. Pour minimiser la distorsion de phase, on privilégiera un filtre RIF, pour réduire la latence, un RII de même ordre. On remarque aussi qu'ajuster la fréquence de coupure permet un meilleur contrôle des harmoniques.

5 Effets audio-numériques

5.1 Effet de réverbération

5.1.1 Expression de l'intercorrélation

On suppose que le signal capté $y(k)$ est le résultat de la convolution du signal d'excitation $x(k)$ avec la réponse impulsionnelle $h(k)$ de la pièce :

$$y(k) = (x * h)(k)$$

L'intercorrélation entre y et x est donc :

$$R_{yx}(u) = (h * R_{xx})(u)$$

5.1.2 Estimation de la réponse impulsionnelle

On utilise un signal d'excitation qui a des propriétés proches d'une impulsion, comme du bruit blanc ou certaines séquences spécifiques. Ces signaux sont appelés pseudo-impulsifs, car leur autocorrélation ressemble à une impulsion de Dirac $\delta(u)$: elle est forte uniquement au centre ($u = 0$) et très faible ailleurs.

Si on utilise un tel signal $x(k)$, alors son autocorrélation est proche de :

$$R_{xx}(u) \approx \delta(u)$$

En partant de la relation :

$$R_{yx}(u) = h(u) * R_{xx}(u)$$

on obtient, par approximation :

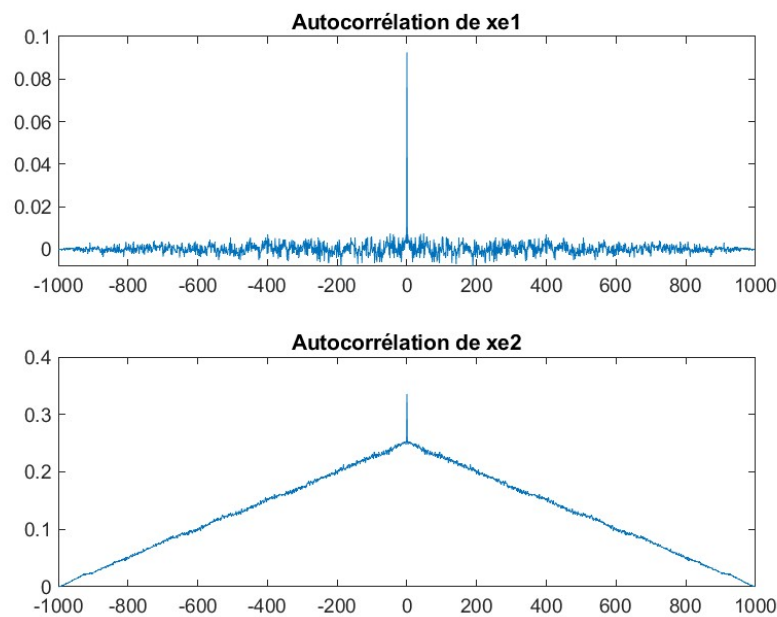
$$R_{yx}(u) \approx h(u)$$

Pour estimer la réponse impulsionnelle $h(k)$, on peut donc calculer l'intercorrélation entre le signal reçu $y(k)$ et le signal d'excitation $x(k)$.

$$h(k) \approx \text{xcorr}(y, x)$$

5.1.3 Choix du meilleur signal d'excitation

On trace les autocorrélation des signaux x_{e1} et x_{e2} . On choisira le signal dont l'autocorrélation est la plus proche d'un pic en 0 comme signal d'excitation pour mesurer la réponse impulsionnelle de la pièce.

FIGURE 9 – Autocorrélations des signaux d'excitation x_{e1} et x_{e2} .

On choisit le signal x_{e1} car il présente une autocorrélation centrée autour de 0 (pic très fin et valeurs négligeables ailleurs). Cela correspond au comportement attendu d'un signal pseudo-impulsif, dont l'autocorrélation est proche d'un Dirac $\delta(u)$.

Cette propriété fait que l'intercorrélation $R_{yx}(u)$ donne une bonne approximation de la réponse impulsionnelle $h(u)$. En comparaison, le signal x_{e2} présente une autocorrélation beaucoup plus étalée et en forme de triangle, et est donc moins adapté pour estimer la réponse impulsionnelle de la pièce.

5.1.4 Estimation de la réponse impulsionnelle d'une pièce

Pour estimer la réponse impulsionnelle d'une pièce, nous avons utilisé le signal x_{e1} , dont l'autocorrélation est proche d'une impulsion (voir Q3.3). Ce signal a été injecté dans une simulation acoustique via la fonction `simule_piece`, qui modélise le comportement réverbérant de la pièce.

La réponse impulsionnelle $h(k)$ a ensuite été estimée en calculant l'intercorrélation $R_{yx}(u)$ entre le signal capté $y(k)$ et le signal d'entrée $x(k)$, puis en extrayant la partie causale (pour $k \geq 0$).

La figure suivante montre la réponse impulsionnelle estimée :

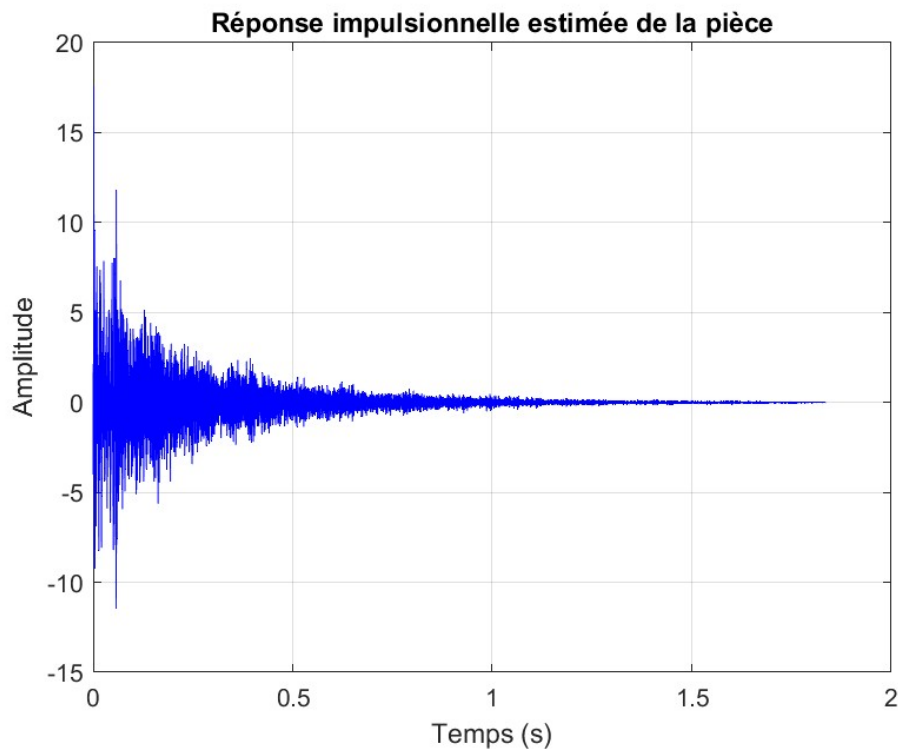


FIGURE 10 – Réponse impulsionnelle estimée de la pièce.

La courbe obtenue présente un pic initial suivi d'une série de réflexions secondaires décroissantes. Cette décroissance est due à l'absorption des parois et à la perte de cohérence des fronts d'onde.

5.1.5 Implémentation d'un effet de réverbération par convolution

La fonction `effet_reverb` est disponible dans les annexes. Elle applique un effet de réverbération à un signal audio en réalisant une convolution entre le signal d'entrée $x(k)$ et la réponse impulsionnelle estimée $h(k)$ de la pièce. Cela simule ainsi la manière dont un son se propage et se réfléchit dans un espace réverbérant.

5.1.6 Test de la fonction `effet_reverb` et mesure du temps d'exécution

On teste notre fonction `effet_reverb` grâce au script MATLAB `test_effet_reverb` sur le son de guitare en nylon (note a3). On utilise la réponse impulsionnelle estimée dans la question 3.4. Les commandes `tic` et `toc` sont utilisées pour chronométrer le temps d'exécution de notre test. Pour ce test, on compare le signal d'origine au signal convolué et "réverbéré".

Dans la première courbe, on observe le signal original de la guitare. Ensuite, sur la courbe du bas, on remarque un signal légèrement plus long et qui semble présenter des échos successifs. À l'écoute, on entend bien un écho ou une réverbération du son.

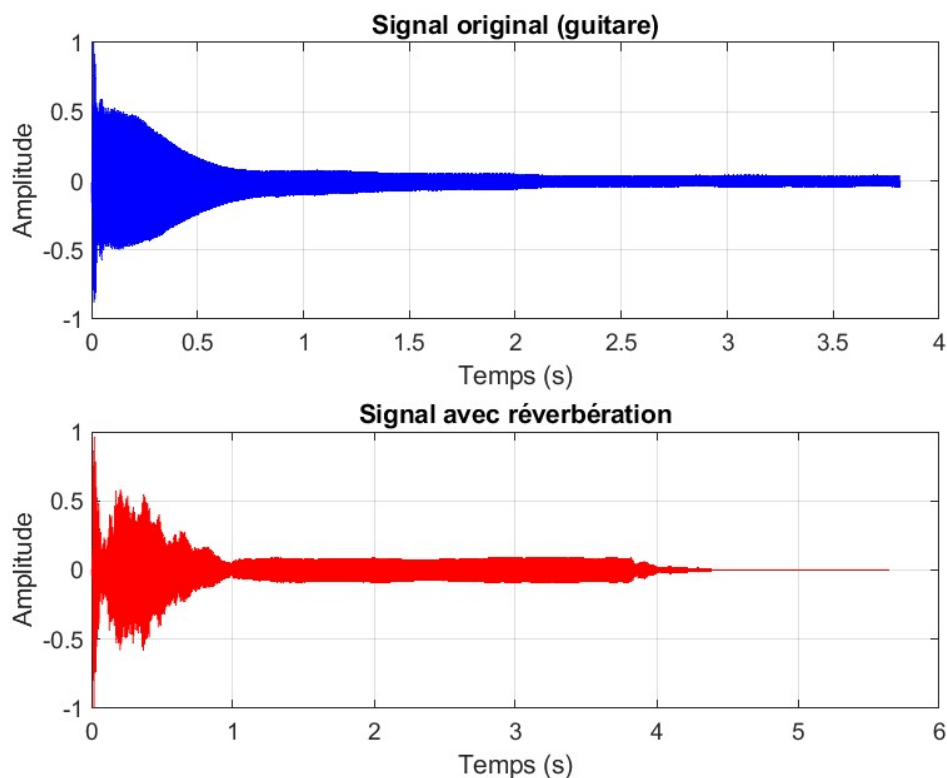


FIGURE 11 – Effet de la réverbération sur un signal de guitare – comparaison des signaux avant et après convolution.

Il semble donc que ce test valide la bonne exécution de la fonction `effet_reverb`. De plus, le temps d'exécution de la fonction `effet_reverb` pour un signal de guitare de 3.8 secondes est d'environ **0.451 secondes** (récupéré grâce à `tic` et `toc`)

5.1.7 Implémentation de la réverbération par convolution fréquentielle

Ici, on remplace la méthode pour effectuer un effet de réverbération : on effectue maintenant l'opération suivante à la place de la convolution :

$$y(k) = \text{IFFT}(\text{FFT}(h(k)) \cdot \text{FFT}(x(k)))$$

où $x(k)$ est le signal source (par exemple, un son de guitare), $h(k)$ la réponse impulsionnelle estimée de la pièce, et $y(k)$ le signal réverbéré. Cette méthode est plus rapide grâce à la complexité algorithmique avantageuse de la transformée de Fourier rapide (FFT).

Nous avons donc codé cette seconde méthode dans (`effet_reverb_FFT.m`) et nous l'avons testé dans (`test_effet_reverb_fft`).

On observe que le résultat est similaire à celui obtenu par convolution, ce qui est attendu puisque les deux approches sont mathématiquement équivalentes (à conditions numériques égales). A l'oreille, le son est aussi réverbéré.

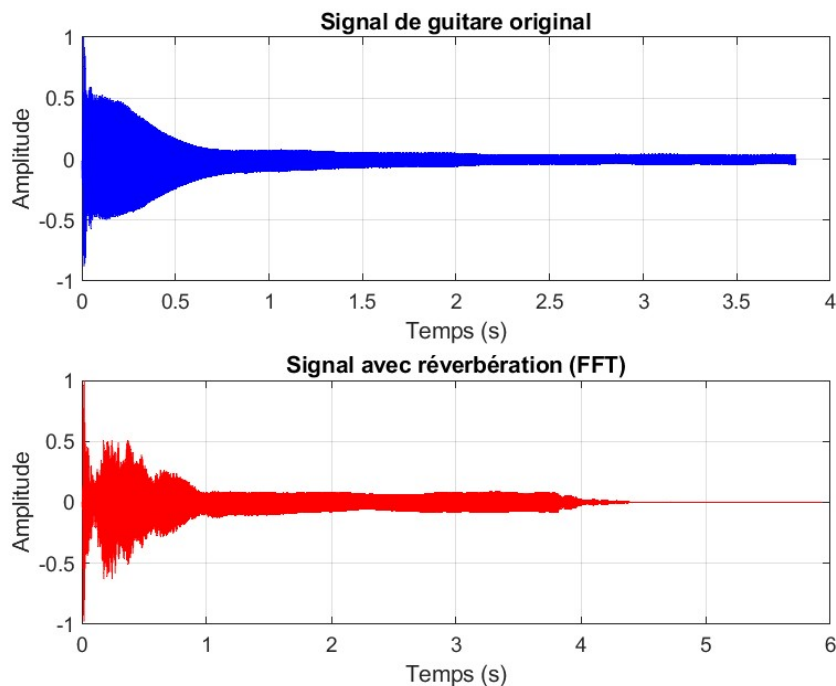


FIGURE 12 – Comparaison entre le signal de guitare original et le signal réverbéré par convolution fréquentielle

Le temps d'exécution mesuré avec les fonctions `tic` et `toc` est d'environ 0.1 secondes à la première exécution et d'environ 0.01 secondes aux exécutions suivantes, contre environ 0.45 secondes pour la première méthode.

Ainsi, la méthode par FFT permet d'obtenir une réverbération identique perceptuellement à celle par convolution, tout en divisant significativement le temps de calcul.

5.1.8 Comparaison entre convolution classique et convolution par FFT

Nous avons testé deux méthodes pour appliquer un effet de réverbération à un signal audio :

- la convolution classique dans le domaine temporel.
- la convolution fréquentielle via la relation :

$$y(k) = \text{IFFT}(\text{FFT}(h(k)) \cdot \text{FFT}(x(k)))$$

Mathématiquement, les deux méthodes sont équivalentes si la transformée de Fourier est calculée avec une longueur suffisante pour éviter le repliement spectral, c'est-à-dire que la FFT est réalisée sur une taille $N \geq L_x + L_h - 1$, où L_x et L_h sont les longueurs respectives des signaux $x(k)$ et $h(k)$. Cette condition garantit que la convolution circulaire par la FFT est équivalente à la convolution linéaire classique.

Dans nos expériences :

- Les signaux de sortie obtenus avec les deux méthodes sont visuellement et auditivement identiques.
- Le temps d'exécution est significativement plus court avec la FFT (0.01 s contre 0.45 s).

En conclusion, La méthode utilisant la FFT est équivalente à la convolution classique à condition d'utiliser un NFFT suffisant pour éviter les effets de bord. De plus, elle est nettement plus rapide et donc mieux adaptée aux applications temps réel ou aux traitements sur des fichiers longs.

5.2 Effet de retard

5.2.1 Réponse impulsionnelle théorique du système à délai

Le système de delay est défini par l'équation suivante :

$$y(k) = x(k) - g \cdot y(k - \tau) \quad (1)$$

où τ est un entier strictement positif représentant un retard en nombre d'échantillons, et g est un coefficient réel d'atténuation.

On cherche la réponse impulsionnelle $h(k)$ de ce système, c'est-à-dire la sortie $y(k)$ lorsque l'entrée est une impulsion unité :

$$x(k) = \delta(k)$$

On résout donc :

$$h(k) = \delta(k) - g \cdot h(k - \tau)$$

Pour $k = 0$:

$$h(0) = \delta(0) - g \cdot h(-\tau) = 1 - 0 = 1$$

Pour $k = \tau$:

$$h(\tau) = \delta(\tau) - g \cdot h(0) = 0 - g \cdot 1 = -g$$

Pour $k = 2\tau$:

$$h(2\tau) = \delta(2\tau) - g \cdot h(\tau) = 0 - g \cdot (-g) = g^2$$

Pour $k = 3\tau$:

$$h(3\tau) = \delta(3\tau) - g \cdot h(2\tau) = 0 - g \cdot g^2 = -g^3$$

Par généralisation, on obtient :

$$h(k) = \begin{cases} (-g)^n & \text{si } k = n\tau \text{ pour un entier } n \geq 0 \\ 0 & \text{sinon} \end{cases}$$

Ainsi, la réponse impulsionnelle du système est une suite échantillonnée tous les τ instants, avec un coefficient décroissant en puissance de $-g$.

5.3 Condition de stabilité du filtre de delay

Pour que le filtre soit stable au sens entrée bornée - sortie bornée, il faut que :

$$\sum_{k=-\infty}^{+\infty} |h(k)| < \infty$$

Cette condition revient à étudier la convergence de la série :

$$\sum_{n=0}^{\infty} |(-g)^n| = \sum_{n=0}^{\infty} |g|^n$$

Or, c'est une série géométrique qui converge si et seulement si $|g| < 1$.

Ainsi, le filtre est stable si et seulement si :

$$|g| < 1$$

5.4 Détermination des vecteurs a et b

L'équation du filtre est donnée par :

$$y(k) = x(k) - g \cdot y(k - \tau)$$

On la réécrit sous forme canonique, en mettant tous les termes en y à gauche :

$$y(k) + g \cdot y(k - \tau) = x(k)$$

Cette écriture permet d'identifier les vecteurs de coefficients nécessaires pour la fonction `filter(b, a, x)` dans MATLAB.

Pour le vecteur b (non-récuratif) : Le terme $x(k)$ est pris tel quel, donc :

$$b = [1]$$

Pour le Vecteur a (récuratif) : Le terme $y(k)$ a un coefficient 1, et $y(k - \tau)$ a un coefficient g . Tous les autres coefficients sont nuls :

$$a = [1, 0, 0, \dots, 0, g] \quad (\text{avec } \tau - 1 \text{ zéros entre 1 et } g)$$

Par exemple, si $\tau = 3$ et $g = 0,7$, on a :

$$a = [1, 0, 0, 0, 0, 7], \quad b = [1]$$

Ce filtre est donc un filtre IIR (à réponse impulsionnelle infinie) capable de modéliser un effet de retard avec une atténuation contrôlée par le coefficient g .

5.4.1 Réponse impulsionnelle du filtre delay (simulation)

On simule la réponse impulsionnelle du filtre delay en lui mettant une impulsion unité entrée. Pour rappel, le filtre est défini par :

$$y(k) = x(k) - g \cdot y(k - \tau) \tag{2}$$

avec g le coefficient de feedback et τ le retard exprimé en échantillons. Ainis, pour simuler la réponse impulsionnelle, on applique la fonction `filter` sur un vecteur contenant une impulsion unité (c'est-à-dire $x(k) = \delta(k)$).

Nous avons choisi les paramètres suivants : $g = 0,6$, $\tau = 3$ et une longueur d'observation $N = 100$.

Le coefficient g détermine l'atténuation appliquée à chaque répétition du signal dans la boucle de feedback. Une valeur de $g = 0,6$ permet de visualiser clairement la décroissance géométrique de l'amplitude sans que le signal ne s'annule trop rapidement (ce qui serait le cas avec un g plus petit) ni ne persiste trop longtemps (ce qui arriverait si g était proche de 1).

Le retard $\tau = 3$ a été choisi pour espacer suffisamment les impulsions dans le signal de sortie tout en gardant un affichage lisible. Ce paramètre contrôle la distance entre les pics successifs dans la réponse impulsionnelle.

Enfin, la taille du vecteur $N = 100$ est suffisante pour observer plusieurs répétitions du signal retardé, ce qui permet de confirmer la nature périodique et décroissante du filtre de delay.

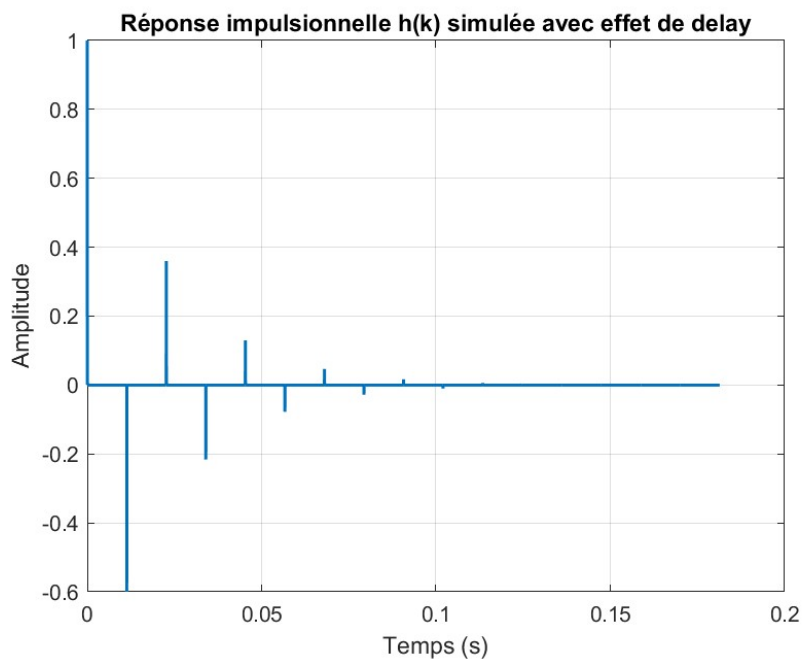


FIGURE 13 – Réponse impulsionnelle $h(k)$ simulée avec effet de delay (via filter)

On observe une série d'impulsions espacées régulièrement toutes les τ échantillons, et dont l'amplitude suit la loi $(-g)^n$. Cette alternance de signes est caractéristique du choix $g < 0$. Le signal décroît exponentiellement en amplitude. Ainsi, la simulation est en cohérence avec la formule théorique de la réponse impulsionnelle donnée à la question 3.9 :

$$h(k) = \begin{cases} (-g)^n & \text{si } k = n\tau \\ 0 & \text{sinon} \end{cases}$$

5.4.2 Réponse en fréquence du filtre de delay

On rappelle l'équation définissant notre filtre :

$$y(k) = x(k) - g \cdot y(k - \tau) \quad (3)$$

Pour chaque terme, on prends la Transformée de Fourier en Temps Discret (DTFT). En utilisant les pro-

propriétés de linéarité et de décalage, on obtient :

$$Y(e^{j2\pi\nu}) = X(e^{j2\pi\nu}) - g \cdot Y(e^{j2\pi\nu})e^{-j2\pi\nu\tau} \quad (4)$$

ensuite, on regroupe les termes en $Y(e^{j2\pi\nu})$:

$$Y(e^{j2\pi\nu}) (1 + ge^{-j2\pi\nu\tau}) = X(e^{j2\pi\nu}) \quad (5)$$

on a alors :

$$H(e^{j2\pi\nu}) = \frac{Y(e^{j2\pi\nu})}{X(e^{j2\pi\nu})} = \frac{1}{1 + ge^{-j2\pi\nu\tau}} \quad (6)$$

On retrouve donc la réponse en fréquence :

$$\hat{h}(\nu) = \frac{1}{1 + ge^{-j2\pi\nu\tau}} \quad (7)$$

On calcule ensuite le module :

$$|\hat{h}(\nu)| = \left| \frac{1}{1 + ge^{-j2\pi\nu\tau}} \right| = \frac{1}{\sqrt{1 + 2g \cos(2\pi\nu\tau) + g^2}} \quad (8)$$

Ce module atteint ses maximums lorsque le terme au dénominateur est minimal, c'est-à-dire lorsque :

$$\cos(2\pi\nu\tau) = -1 \Rightarrow 2\pi\nu\tau = (2k + 1)\pi \Rightarrow \nu_k = \frac{2k + 1}{2\tau} \quad (k \in \mathbb{Z}) \quad (9)$$

Dans ce cas, on obtient :

$$|\hat{h}(\nu_k)| = \frac{1}{1 - g} \quad (10)$$

Pour la phase, on étudie l'argument complexe :

$$\arg(\hat{h}(\nu)) = -\arg(1 + ge^{-j2\pi\nu\tau}) = -\tan^{-1} \left(\frac{g \sin(2\pi\nu\tau)}{1 + g \cos(2\pi\nu\tau)} \right) \quad (11)$$

Ainsi, la réponse en fréquence du filtre de delay est entièrement déterminée par le paramètre de gain g , la valeur du retard τ , et la fréquence réduite ν .

Pour conclure, la réponse en fréquence du filtre forme un filtre en peigne : elle présente une structure répétée dans le domaine fréquentiel avec des pics (ou creux) réguliers, typiques des délais.

5.4.3 Comparaison entre réponse en fréquence théorique et numérique

On cherche ici à comparer la réponse en fréquence théorique du filtre de delay avec celle obtenue numériquement via la transformée de Fourier discrète (DFT) de sa réponse impulsionnelle.

Pour ce faire, nous avons utilisé les paramètres suivants :

$g = 0,8$: ce coefficient d'atténuation relativement élevé permet d'obtenir des résonances marquées dans la réponse fréquentielle ; $\tau = 20$: un retard important permet d'observer un motif répétitif fin dans le domaine fréquentiel ; $N = 1000$: la taille de la DFT est suffisamment grande pour offrir une bonne résolution fréquentielle ; $F_e = 44100$ Hz : fréquence d'échantillonnage classique pour les signaux audio.

Sur la figure obtenue, on peut voir une superposition presque parfaite entre la courbe théorique (en bleu) et la courbe obtenue numériquement (en rouge pointillé). On observe de plus une structure en peigne.

Les pics apparaissent aux fréquences réduites de la forme :

$$\nu_k = \frac{2k+1}{2\tau}, \quad k \in \mathbb{Z}$$

et leur amplitude maximale est donnée par :

$$|H(\nu_k)| = \frac{1}{1-g}$$

Cependant, le fait que la simulation corresponde quasiment parfaitement à la simulation montre que le choix de $N = 1000$ est suffisant pour capturer les résonances de ce filtre IIR : En effet, on pourrait avoir éventuellement des différences avec un autre choix de valeur de N qui pourraient provenir de la troncature de la réponse impulsionnelle ou des effets de bord.

Ainsi on peut valider que la réponse en fréquence théorique du filtre de delay est bien retrouvée numériquement.

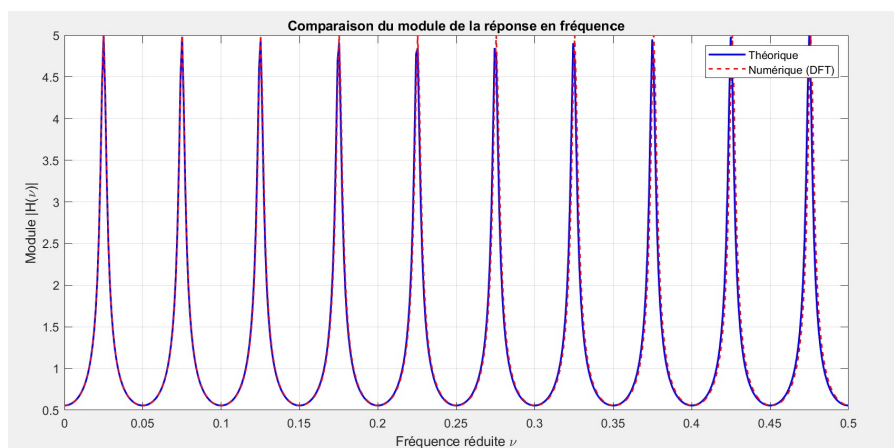


FIGURE 14 – Comparaison entre le module de la réponse en fréquence théorique et celui estimé numériquement (DFT) du filtre à retard.

5.4.4 Implémentation d'un effet de delay simple (IIR)

La fonction `effet_delay` est disponible en annexe. Elle ajoute une effet de delay (retard) à un son en prenant le signal d'entrée et en y ajoutant une version décalée.

Elle commence par convertir le temps de retard en nombre d'échantillons. Ensuite, elle parcourt le signal, et à chaque point, elle combine le son actuel avec une version plus ancienne du même son, atténuée selon un facteur g .

5.4.5 Test de l'effet de delay simple

On teste maintenant la fonction `effet_delay` sur le fichier audio `piano1` (voir code q3_16).

Le fichier en sortie présente un effet d'écho clair, avec des répétitions. Les répétitions sont progressivement atténuées en intensité.

5.4.6 Effet de delay avec filtrage dans la boucle de retour

La fonction `effet_delay_filtre` ajoute un effet d'écho à un audio

Cependant ici, pour que l'écho soit plus naturel et moins "brut", on fait une moyenne des anciennes valeurs du signal avant de les réinjecter. Cela rend le son plus "doux" à l'écoute.

5.4.7 Test de l'effet de delay filtré

Nous avons appliqué la fonction `effet_delay_filtre` sur un son de piano avec un délai de 0,25 s, un coefficient d'amortissement $g = 0,9$ et une moyenne glissante de longueur $K = 10$ dans la boucle de feedback. Le fichier de sortie, nommé `piano_delay_filtre.wav`, contient un son avec un effet d'écho plus doux que dans la version sans filtrage. A l'oreille, la différence est très subtile et faiblement distinguable.

5.4.8 Justification de l'atténuation des hautes fréquences par le filtre $h_r(k)$

Nous avons tracé le module de la réponse en fréquence du filtre $h_r(k)$ utilisé dans la boucle de feedback du delay filtré. Ce filtre correspond à une moyenne glissante sur $K = 10$ échantillons.

La figure montre que le filtre agit comme un passe-bas : les basses fréquences sont peu atténuées, tandis que les hautes fréquences sont progressivement réduites. Cela explique le fait qu'on entende un son qui semble couper les hautes fréquences.

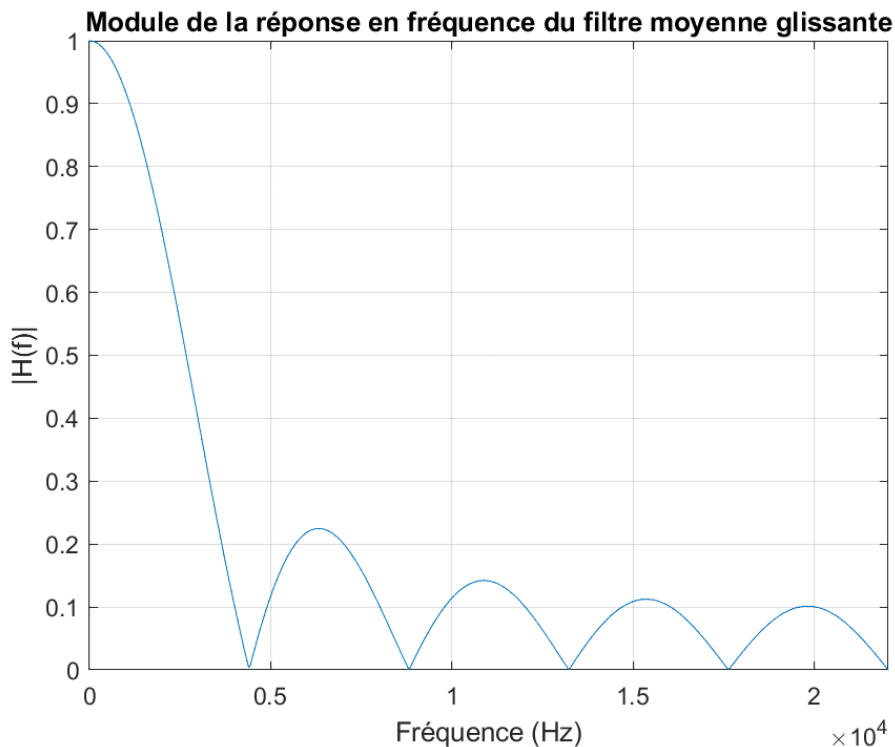


FIGURE 15 – Module de la réponse en fréquence du filtre moyenne glissante ($K = 10$).

Conclusion

Ce TP nous a permis d'étudier la synthèse additive et soustractive, ainsi que des effets audio comme la réverbération et le delay. L'analyse des harmoniques et l'application des enveloppes ADSR ont amélioré le réalisme des sons. Les méthodes fréquentielles se sont avérées plus efficaces que les temporelles. Les filtres et retards ont démontré leur utilité pour créer des effets sonores. Enfin, les simulations qu'on avait ont confirmé nos modèles théoriques.