

# SAR - Traitement audio

Medjibe HASHIM-FAROOQI , Titouan LHOMME

May 21, 2025

## Introduction

Ce projet a pour objectif d'illustrer concrètement les notions fondamentales du traitement du signal à travers une série d'expérimentations numériques réalisées sous MATLAB. Il s'appuie sur des outils et concepts vus en cours, tout en mettant l'accent sur leur utilité dans l'analyse et la transformation de signaux réels, en particulier dans le domaine audio.

Après une brève présentation des fonctions MATLAB utilisées tout au long du travail, nous nous intéressons à la problématique de l'échantillonnage et à ses implications pratiques, en mettant en œuvre le théorème de Shannon-Nyquist. L'analyse fréquentielle des signaux est ensuite abordée à travers la transformée de Fourier discrète, permettant de passer de la représentation temporelle à une lecture plus informative dans le domaine fréquentiel.

Enfin, le projet se concentre sur l'implémentation de systèmes linéaires invariants dans le temps pour concevoir des effets audio-numériques. L'effet de delay, la convolution et les filtres sont explorés à la fois sur le plan théorique et par des expérimentations sonores, révélant leur impact sur la perception auditive. L'ensemble du travail vise ainsi à faire le lien entre formalisme mathématique, implémentation algorithmique et écoute concrète.

## 1 Synthèse additive

### 1.1 Analyse d'un son harmonique

Dans un premier temps, on peut comparer quelques instruments pour observer les différents spectres et fréquences fondamentales :

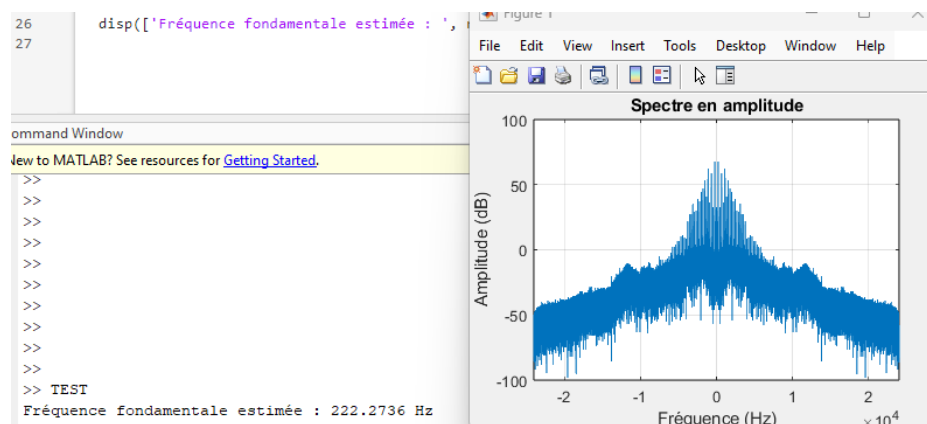


Figure 1: Clarinette

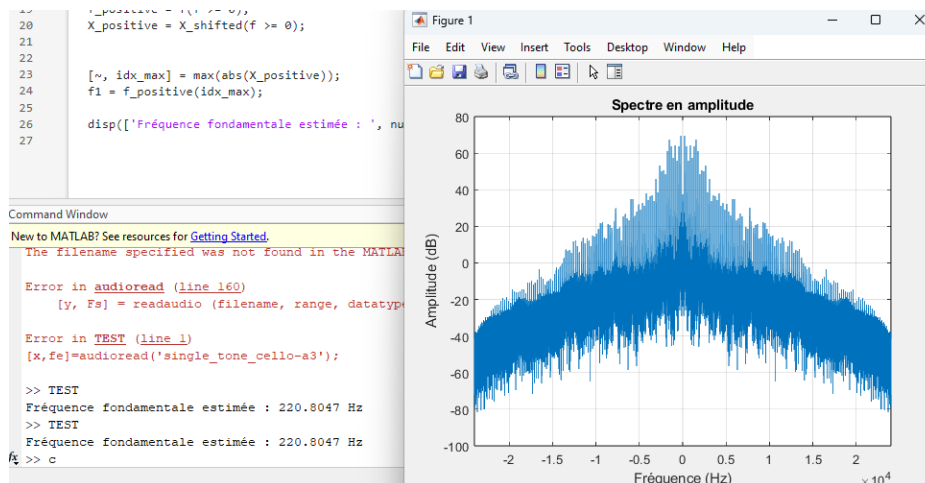


Figure 2: Violoncelle

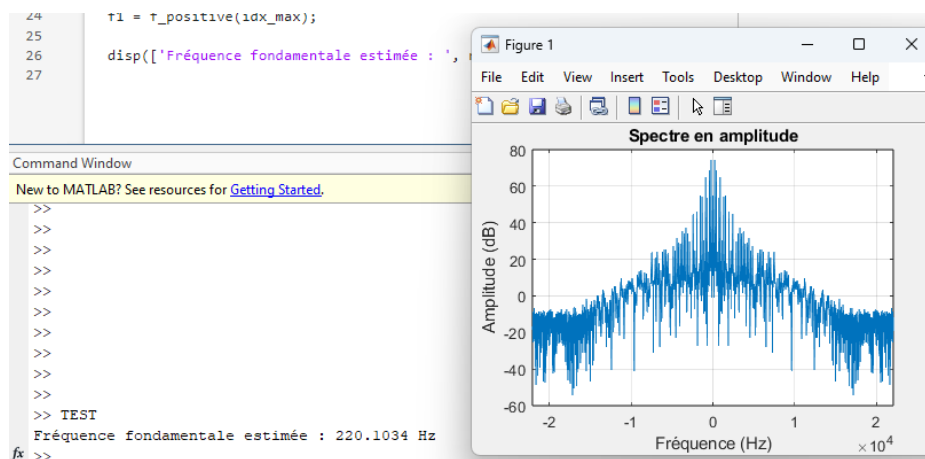


Figure 3: Harpe

## Analyse

Ces instruments ont tous une fréquence fondamentale proche de 220 Hz.

On observe que les instruments à cordes sont quasi-harmonique avec des raies équidistantes de  $f_1$ . De plus, on peut voir que la rigidité des cordes comme sur la harpe implique une légère inharmonicité observée avec des raies moins régulières.

## Harmonicité

Cette fois ci, en se basant sur les informations à disposition, on trace sur un même graphe les spectres des deux signaux. On analyse ensuite le tableau fourni par le code.

En effet, plus les valeurs sont proches de 0, plus le piano est harmonieux. De plus, un piano avec des très variables ou loin de 0 est plus inharmonique.

$n$	1	2	3	4	5	6	7	8
$\xi_n$ piano 1 [c]	0	0	$2.1 \times 10^{-12}$	0	0	$2.1 \times 10^{-12}$	0	0
$\xi_n$ piano 2 [c]	0	0	0	0	0	0	0	0

Figure 4: Comparaison des 2 signaux

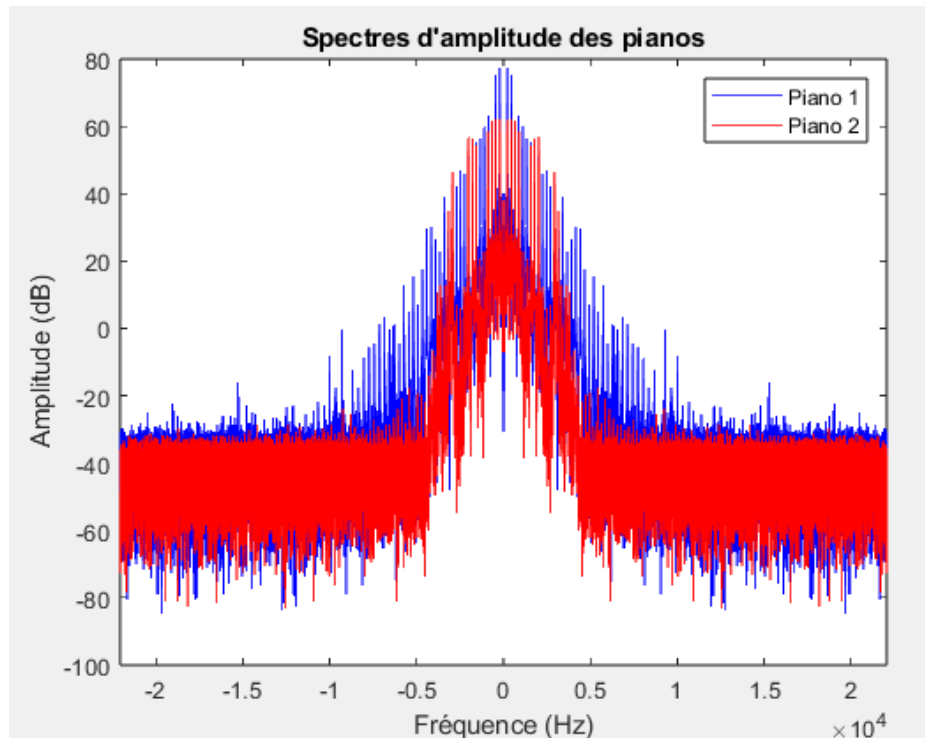


Figure 5: Comparaison des 2 signaux

Dans notre cas, le signal 2 est le plus harmonique

*→ écouté le, vous devriez trouver que votre conclusion n'est peut être pas bonne...*

## 1.2 Comparaison des méthodes de synthèse sonore

Dans cette étude, nous avons synthétisé un son de piano à partir de ses huit premières composantes harmoniques à l'aide de trois approches différentes. Chaque méthode vise à reproduire les caractéristiques sonores d'un piano réel avec plus ou moins de fidélité temporelle et spectrale.

### 1.2.1 Superposition de sinusoïdes (synthèse additive)

Cette méthode consiste à sommer huit sinusoïdes correspondant aux fréquences mesurées dans le spectre du piano. Chaque harmonique est modélisée par une onde sinusoïdale de fréquence  $f_n$  et d'amplitude extraite du spectre.

**Résultat :**

- Son clair et bien défini, reproduisant les harmoniques.
- Cependant, le signal reste statique et artificiel.
- L'absence d'évolution temporelle rend le son peu réaliste.

*? ça veut dire quoi?*

### 1.2.2 Superposition + enveloppe ADSR

Ici, une enveloppe ADSR (Attack, Decay, Sustain, Release) est appliquée au signal obtenu précédemment pour simuler la dynamique naturelle d'un instrument acoustique.

**Résultat :**

- Le son gagne en réalisme.
- La dynamique temporelle (attaque rapide, décroissance, relâchement) reproduit mieux le comportement d'un piano.
- Le timbre est perçu comme plus naturel et agréable.

*il n'a pourtant pas changé...*

*vous n'avez pas fourni de source pour que je comprenne ce que vous avez fait avec tout vos trucs vos trucs conclusions*

### 1.2.3 Synthèse via transformée de Fourier inverse (IFFT)

Dans cette approche, un spectre est construit en insérant les harmoniques dans le domaine fréquentiel. La transformée de Fourier inverse est ensuite utilisée pour reconstruire le signal temporel.

**Résultat :**

- Signal quasi identique à la méthode 1 (superposition simple).
- Très fidèle au contenu spectral, mais tout aussi figé sans enveloppe.
- L'ajout d'une enveloppe ADSR permet de retrouver une qualité équivalente à la méthode 2.

### 1.3 Comparatif des trois méthodes

Méthode	Rendu sonore	Dynamique	Fidélité spectrale	Appréciation
Sinusoïdes simples	Faible	Faible	Élevée	Artificiel, monotone
Sinusoïdes + ADSR	Élevé	Réaliste	Élevée	Réaliste, expressif
IFFT seule	Faible	Faible	Élevée	Identique à sinusoïdes simples
IFFT + ADSR	Élevé	Réaliste	Élevée	Comparable à méthode 2

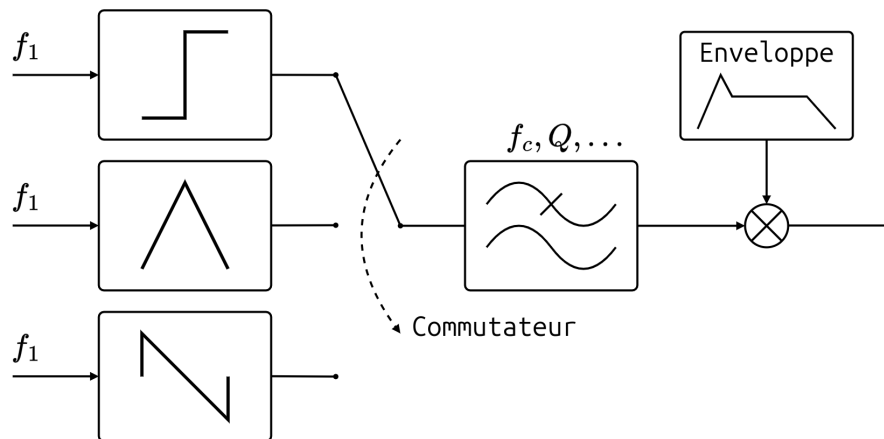
Table 1: Comparaison des différentes approches de synthèse sonore

### Conclusion

Parmi les méthodes testées, la synthèse additive enrichie par une enveloppe ADSR est celle qui offre le meilleur compromis entre simplicité, réalisme et fidélité spectrale. La méthode IFFT constitue une alternative mathématiquement équivalente, à condition elle aussi d'inclure une modulation temporelle.

## 2 Spectre de signaux périodiques pour la synthèse soustractive

Dans la section précédente, la génération d'une note a été réalisée en partant de la fréquence fondamentale enrichie par des harmoniques. Nous abordons ici une autre technique : partir d'un signal périodique possédant naturellement des harmoniques, que l'on filtre ensuite. C'est le principe utilisé notamment dans le synthétiseur TB-303, célèbre pour son rôle dans le développement de la musique électronique des années 1980. Ce principe est illustré sur la figure suivante :



Considérons un signal carré centré de période  $T$  et d'amplitude  $\pm 1$ . Ce signal est impair et périodique, donc sa série de Fourier ne contient que des sinus impairs :

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

**Remarques :**

OK mais  
vous ne proposez pas  
de méthode pour vérifier  
cela avec Matlab!

- Seules les harmoniques impaires sont présentes.
- L'amplitude de chaque harmonique décroît en  $\frac{1}{n}$ .

Un signal dent de scie centré, d'amplitude  $\pm 1$  et de période  $T$ , possède une série de Fourier contenant toutes les harmoniques (paires et impaires). Si on le définit linéairement de  $-1$  à  $1$  sur la période, il est impair :

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

#### Remarques :

- Toutes les harmoniques sont présentes.
- L'amplitude décroît en  $\frac{1}{n}$ , avec alternance de signe.

Pour vérifier ces spectres, on peut échantillonner le signal sur plusieurs périodes (durée multiple de  $T$ ). Puis appliquer la transformée de Fourier discrète (FFT) pour obtenir le spectre numérique et enfin, comparer les amplitudes des pics spectraux à celles des coefficients analytiques.

Les limites de la méthode numérique sont la résolution fréquentielle qui dépend de la durée totale échantillonnée, l'aliasing car si la fréquence d'échantillonnage est insuffisante, le fenêtrage car l'absence de fenêtre (troncature brutale) introduit des artefacts (effet de Gibbs) et la précision numérique car les composantes faibles peuvent être noyées dans le bruit de quantification.

Maintenant, considérons un filtre passe-bas d'ordre 1 défini par :

$$y(k) = 0,5 \cdot (x(k) + x(k-1)).$$

Ce filtre est un FIR non récursif d'ordre 1, avec :

$$b = [0,5 \quad 0,5], \quad a = 1,$$

et en utilisant MATLAB avec :

`y = filter(b, a, x);`

La réponse impulsionnelle est :

$$h(n) = \begin{cases} 0,5 & n = 0 \\ 0,5 & n = 1 \\ 0 & \text{sinon} \end{cases}$$

Sa transformée de Fourier est :

$$H(e^{j\omega}) = 0,5 + 0,5e^{-j\omega} = 0,5(1 + e^{-j\omega}).$$

En utilisant l'identité d'Euler :

$$H(e^{j\omega}) = e^{-j\omega/2} \cdot \cos\left(\frac{\omega}{2}\right).$$

Le module vaut donc :

$$|H(e^{j\omega})| = |\cos(\omega/2)|.$$

Ce filtre a un gain maximal égale à 1 à  $\omega = 0$ , et un gain nul à  $\omega = \pi$  (fréquence de Nyquist).

On peut appliquer ce filtre à un signal périodique et observer la réduction progressive des harmoniques élevées conformément à  $|\cos(\omega/2)|$ .

#### Observations :

- Les basses fréquences (fondamentale) sont peu atténuées.
- Les hautes fréquences (harmoniques) sont atténuées progressivement.
- Le filtrage adoucit le spectre.

*=> Vérification avec Matlab?*

Méthode	Source	Timbre	Spectre	Remarques
Soustractive	Carré filtré	Doux, un peu creux	Moyennement riche	Filtrage adoucit un spectre simple (harmoniques impaires)
Soustractive	Dent de scie filtrée	Chaud, brillant	Très riche	Spectre complexe partiellement atténué, son proche d'un instrument
Additive	Carré reconstruit	Synthétique, net	Moyennement riche	Fidèle mathématiquement, mais peu naturel
Additive	Dent de scie reconstruite	Riche, parfois agressif	Très riche	Spectre complet, son brut sans filtrage

Table 2: Comparaison des sons produits par synthèse soustractive et additive (avec la même enveloppe ADSR)

### Comparaison synthèse soustractive vs additive

**Conclusion :** La synthèse soustractive offre des sons plus organiques et évolutifs grâce au filtrage dynamique. La synthèse additive permet un contrôle très fin du contenu fréquentiel, mais produit souvent des sons plus artificiels. Et utiliser la même enveloppe ADSR permet une comparaison équitable du timbre uniquement.

Pour améliorer la qualité sonore de la synthèse soustractive, on peut utiliser un filtre plus sophistiqué qu'un simple passe-bas d'ordre 1.

Exemple de création d'un filtre FIR passe-bas d'ordre 2 avec fréquence de coupure normalisée à 0,25 :

```
d = designfilt('lowpassfir', 'FilterOrder', 2, 'HalfPowerFrequency', 0.25);
```

On peut analyser ce filtre avec :

```
filterAnalyzer(d);
```

Et filtrer un signal par :

```
y = filter(d, x);
```

#### Améliorations possibles :

- Augmenter l'ordre du filtre car un ordre plus élevé (ex. 20, 50) donne une pente plus raide et un filtrage plus net des harmoniques.
- Modifier la fréquence de coupure car ajuster la fréquence pour contrôler le timbre (coupure basse = son plus sombre).
- Changer la nature du filtre car passer à un filtre IIR (Butterworth, Chebyshev) pour des transitions plus nettes ou des effets de résonance.

**Conclusion** Un filtrage plus avancé permet un contrôle plus précis du spectre et du timbre en synthèse soustractive. Selon les paramètres choisis, on peut obtenir des sons plus doux, plus clairs ou plus expressifs, se rapprochant d'instruments réels ou de textures complexes.

## 3. Effets audio-numériques

### Modélisation de la réverbération par convolution

La réverbération d'une salle peut être modélisée comme un système linéaire invariant dans le temps, complètement caractérisé par sa réponse impulsionnelle  $h(k)$ . Pour simuler cet effet sur un signal audio  $x(k)$ , on peut utiliser la méthode de convolution. Dans le cas de la mesure pseudo-impulsionnelle, on envoie un signal d'excitation  $x(k)$  dans la salle, puis on enregistre la réponse  $y(k)$  captée par un microphone. On cherche alors à estimer la réponse impulsionnelle  $h(k)$  du système.

Cette estimation passe par l'analyse des corrélations entre les signaux. On s'intéresse en particulier à la fonction d'intercorrélation entre  $y$  et  $x$ , notée  $R_{yx}(u)$ , ainsi qu'à la fonction d'autocorrélation de  $x$ , notée  $R_{xx}(u)$ .

En supposant que la réponse  $y(k)$  est obtenue par la convolution de  $x(k)$  avec  $h(k)$ , c'est-à-dire :

$$y(k) = \sum_m h(m)x(k-m),$$

on peut exprimer la fonction d'intercorrélation  $R_{yx}(u)$  comme suit :

$$\begin{aligned} R_{yx}(u) &= \sum_k y(k)x(k+u) \\ &= \sum_k \left( \sum_m h(m)x(k-m) \right) x(k+u) \\ &= \sum_m h(m) \sum_k x(k-m)x(k+u) \\ &= \sum_m h(m) \sum_n x(n)x(n+u+m) \quad (\text{en posant } n = k-m) \\ &= \sum_m h(m)R_{xx}(u+m) \end{aligned}$$

Ainsi, on obtient la relation fondamentale suivante entre l'intercorrélation  $R_{yx}(u)$ , la réponse impulsionnelle  $h(m)$ , et l'autocorrélation  $R_{xx}(u)$  :

$$R_{yx}(u) = \sum_m h(m)R_{xx}(u+m)$$

Cette expression met en évidence que l'intercorrélation entre la sortie et l'entrée est la convolution de la réponse impulsionnelle avec la fonction d'autocorrélation de l'entrée. Cela constitue une base pour l'estimation de  $h(k)$  dans le cadre de la méthode pseudo-impulsionnelle.

Dans le cas idéal, si le signal d'excitation  $x(k)$  possède une fonction d'autocorrélation qui s'apparente à une impulsion unité, c'est-à-dire :

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

alors on peut simplifier l'expression précédemment obtenue pour l'intercorrélation :

$$R_{yx}(u) = \sum_m h(m)R_{xx}(u+m).$$

En remplaçant  $R_{xx}(u)$  par  $\delta(u)$ , on obtient :

$$R_{yx}(u) \approx \sum_m h(m)\delta(u+m).$$

Or, on sait que la convolution avec une impulsion unité décalée sélectionne simplement la valeur de la fonction à l'opposé du décalage :

$$\sum_m h(m)\delta(u+m) = h(-u).$$

Cela nous donne une méthode simple pour estimer la réponse impulsionnelle :

$$\boxed{h(k) \approx R_{yx}(k)}$$

Cette relation signifie que, sous l'hypothèse  $R_{xx}(u) \approx \delta(u)$ , la réponse impulsionnelle  $h(k)$  peut être directement obtenue par l'intercorrélation croisée entre le signal de sortie  $y(k)$  et le signal d'entrée  $x(k)$ , ~~en inversant simplement le signe du décalage~~. Ce principe est à la base de la méthode dite pseudo-impulsionnelle pour la mesure acoustique.

Afin de déterminer lequel des deux signaux `xe1` ou `xe2` est le plus adapté pour estimer la réponse impulsionnelle d'une pièce, nous avons analysé leurs fonctions d'autocorrélation respectives à l'aide de la fonction `xcorr` de MATLAB.

Un bon signal d'excitation pour la mesure pseudo-impulsionnelle doit posséder une fonction d'autocorrélation proche d'une impulsion de Dirac, c'est-à-dire un pic très marqué à l'origine (en  $u = 0$ ) et des valeurs proches de zéro pour tous les autres décalages  $u \neq 0$ .

Cette propriété garantit que le signal n'est pas corrélé avec ses versions décalées, ce qui permet une estimation plus précise de la réponse impulsionnelle à partir de l'intercorrélacion.

Après analyse, le signal **xe1** satisfait bien à ces critères : il présente un pic unique et dominant à l'origine et les valeurs de corrélation pour  $u \neq 0$  sont négligeables.

En revanche, le signal **xe2** montre des fluctuations plus importantes hors de l'origine, ce qui le rend moins adapté pour l'estimation.

**Conclusion :** le signal **xe1** est le plus approprié pour mesurer la réponse impulsionnelle de la pièce.

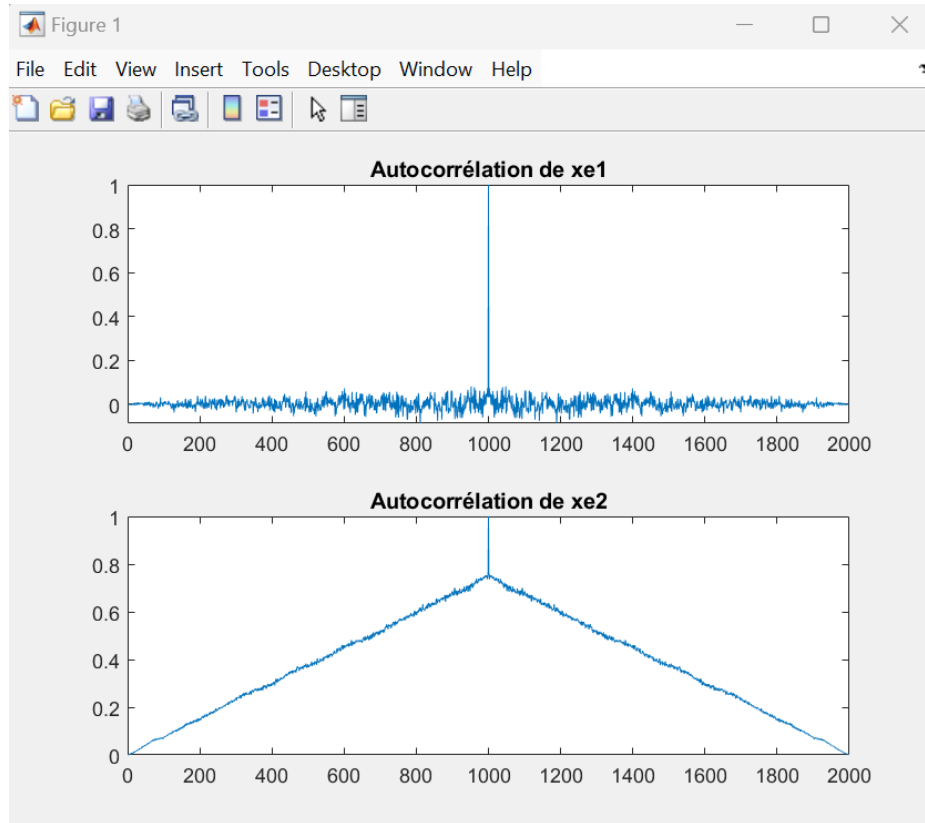


Figure 6: Autocorrélation du signal **xe1**

La suite de l'analyse consiste à estimer la réponse impulsionnelle de la pièce en utilisant le signal d'excitation **xe1** sélectionné précédemment. Ce signal est injecté dans la fonction **simule\_piece**, qui simule l'effet acoustique de la pièce sur le signal d'entrée et génère le signal de sortie  $y(k)$ .

La réponse impulsionnelle  $h(k)$  peut être obtenue par corrélation croisée entre le signal de sortie  $y(k)$  et le signal d'entrée  $x(k)$ , selon la relation suivante :

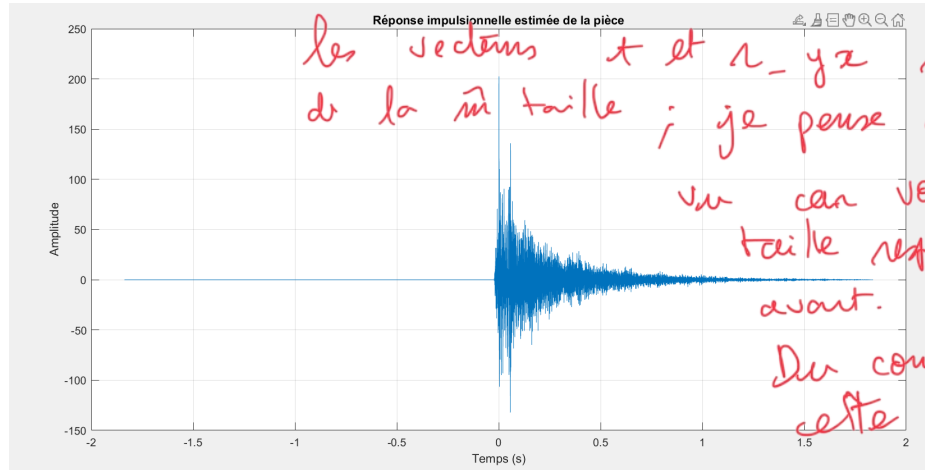
$$h(k) \approx R_{yx}(k) = \sum_n y(n) \cdot x(n+k)$$

Cette méthode repose sur l'approximation  $R_{xx}(k) \approx \delta(k)$ , valable ici puisque **xe1** présente une autocorrélation proche de celle d'une impulsion de Dirac.

La réponse impulsionnelle ainsi estimée est représentée sur la figure suivante. L'axe temporel est exprimé en secondes à l'aide de la fréquence d'échantillonnage  $f_e$  fournie dans le fichier **signal\_excitation.mat**.



Votre script Question 3.3.m produit une erreur et ne trace pas cette figure, car



les vecteurs  $x$  et  $x_{yz}$  ne sont pas de la même taille ; je pense que vous l'avez vu car vous affichez leur taille respective juste avant.

Du coup, d'où vient cette figure ?

Figure 7: Réponse impulsionnelle estimée de la pièce par corrélation croisée

## Application de la réverbération par convolution

Une fois la réponse impulsionnelle  $h(k)$  estimée, il est possible de simuler l'effet de la pièce sur n'importe quel signal audio. Pour cela, on applique une convolution directe entre le signal d'entrée  $x(k)$  et la réponse impulsionnelle  $h(k)$ , ce qui permet de reproduire les effets acoustiques caractéristiques du lieu (réverbérations, échos, etc.).

Nous avons donc implémenté une fonction nommée `effet_reverb`, qui réalise cette opération de convolution directe en utilisant la fonction MATLAB `conv`. Cette fonction prend en entrée le signal audio original et la réponse impulsionnelle, et retourne le signal transformé  $y(k)$  selon la relation suivante :

$$y(k) = \sum_m x(m) \cdot h(k - m)$$

Pour évaluer la performance de cette approche, nous avons appliqué l'effet de réverbération à un signal de guitare à l'aide du script `test_effet_reverb`. Grâce aux commandes `tic` et `toc`, nous avons mesuré le temps de calcul nécessaire à l'exécution de la convolution directe. Ce temps de traitement peut devenir significatif lorsque les signaux sont longs, ce qui soulève la question de l'optimisation de l'algorithme.

Une méthode plus efficace en termes de temps de calcul consiste à réaliser la convolution dans le domaine fréquentiel à l'aide de la transformée de Fourier discrète (DFT). En effet, grâce à la propriété suivante :

$$x(k) * h(k) \longleftrightarrow \mathcal{F}^{-1} [\mathcal{F}(x(k)) \cdot \mathcal{F}(h(k))],$$

on peut appliquer l'effet de réverbération en trois étapes. Premièrement, le calcul de la FFT des deux signaux  $X(f) = \text{fft}(x)$  et  $H(f) = \text{fft}(h)$ . Ensuite, la multiplication point à point  $Y(f) = X(f) \cdot H(f)$ . Enfin, la transformée inverse  $y(k) = \text{ifft}(Y(f))$ .

Nous avons codé cette approche dans une fonction nommée `effet_reverb_FFT`. La taille de la transformée  $N_{\text{FFT}}$  est choisie comme la somme des longueurs des deux signaux moins un, c'est-à-dire :

$$N_{\text{FFT}} = \text{length}(x) + \text{length}(h) - 1,$$

ce qui garantit une convolution linéaire sans repliement fréquentiel (aliasing).

Le temps de calcul de cette version fréquentielle a également été mesuré avec `tic/toc`, et comparé à celui de la version temporelle. Nous avons observé une nette amélioration de la performance, surtout pour les signaux de grande taille : l'utilisation de la FFT réduit considérablement la complexité algorithmique, passant de  $\mathcal{O}(N^2)$  pour la convolution directe à  $\mathcal{O}(N \log N)$  avec la FFT.

Cependant, cette amélioration en vitesse ne doit pas masquer une interrogation importante : la convolution fréquentielle est-elle équivalente à la convolution directe ? En théorie, la réponse est oui, à condition que la taille de la FFT soit suffisante pour éviter le recouvrement circulaire (aliasing temporel),

votre script n'est pas exécutable (fonction définie en début de script)

et la récupération du max ligne 37 n'est pas correct.

Pourquoi la somme ? plus que le max des 2 devrait suffire !

pourquoi ne donnez vous pas les chiffres dans votre rapport ?

OK

non c'est trop !

c'est-à-dire qu'on utilise une FFT de taille au moins égale à  $\text{length}(x) + \text{length}(h) - 1$ . Dans notre implémentation, cette précaution a été prise, ce qui assure l'équivalence des résultats numériques obtenus par les deux méthodes.

**Conclusion :** La réverbération audio peut être efficacement simulée par convolution avec la réponse impulsionnelle d'une pièce. Si la méthode directe donne des résultats corrects, la convolution fréquentielle via FFT permet un gain de performance significatif sans perte de qualité, à condition de bien gérer la taille de la transformée. Cette méthode est donc à privilégier pour les traitements temps réel ou les signaux de longue durée.

## Effet de retard

On considère le système défini par la relation de récurrence suivante :

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

où  $x(k)$  est le signal d'entrée,  $y(k)$  le signal de sortie,  $g$  un coefficient réel d'atténuation, et  $\tau \in \mathbb{N}$  représente un délai en nombre d'échantillons.

## Objectif

Déterminer la réponse impulsionnelle théorique  $h(k)$ , c'est-à-dire la sortie du système pour une entrée impulsionnelle  $x(k) = \delta(k)$ , en supposant un filtre causal.

## Équation à démontrer

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

## Démonstration

On suppose  $x(k) = \delta(k)$ , c'est-à-dire une impulsion de Dirac. On utilise alors la relation de récurrence :

$$y(k) = \delta(k) - g y(k - \tau)$$

On suppose le système **causal**, donc  $y(k) = 0$  pour tout  $k < 0$ .

Calculons les premières valeurs de  $y(k)$  :

- Pour  $k = 0$  :

$$y(0) = \delta(0) - g y(-\tau) = 1 - 0 = 1$$

- Pour  $k = \tau$  :

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

- Pour  $k = 2\tau$  :

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

- Pour  $k = 3\tau$  :

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

Par récurrence, on en déduit que :

$$y(k) = (-g)^n \quad \text{si } k = n\tau$$

et que  $y(k) = 0$  sinon.

## Conclusion

La réponse impulsionnelle théorique du système est donc :

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

ce qui correspond à la formule donnée.

Le filtre défini par l'équation suivante :

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

est un filtre récursif (IIR), car la sortie dépend d'une version retardée d'elle-même.

Pour qu'un filtre IIR soit stable (au sens BIBO : Bounded Input, Bounded Output), il faut que sa réponse impulsionnelle soit absolument sommable, c'est-à-dire :

$$\sum_{k=0}^{\infty} |h(k)| < \infty$$

Or, on a montré que la réponse impulsionnelle est donnée par :

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

La somme des valeurs absolues devient une série géométrique :

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

Cette série converge si et seulement si  $|g| < 1$ .

**Conclusion :** le filtre est stable si et seulement si :

$$\boxed{|g| < 1}$$

On souhaite identifier les vecteurs **a** et **b** pour pouvoir utiliser la fonction standard `filter(b, a, x)` en MATLAB ou Python.

L'équation du filtre est :

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

Cela correspond à un filtre avec :

- Une seule composante dans le numérateur : **b** = [1]
- Une partie récursive avec un retard de  $\tau$  échantillons, ce qui donne :

$$\mathbf{a} = [1, \underbrace{0, \dots, 0}_{\tau-1 \text{ zéros}}, g]$$

**Exemple :** pour  $\tau = 3$ , on a :

$$\mathbf{a} = [1, 0, 0, g] \quad \text{et} \quad \mathbf{b} = [1]$$

On peut tracer la réponse impulsionnelle de ce filtre. On constate que l'amplitude de  $h$  diminue pour  $k$  croissant en alternant en tre valeurs positives et négatives à intervalle de décalage constant. On retrouve bien l'effet de retard que l'on étudie qui s'appliquent à certaines fréquences du signal.

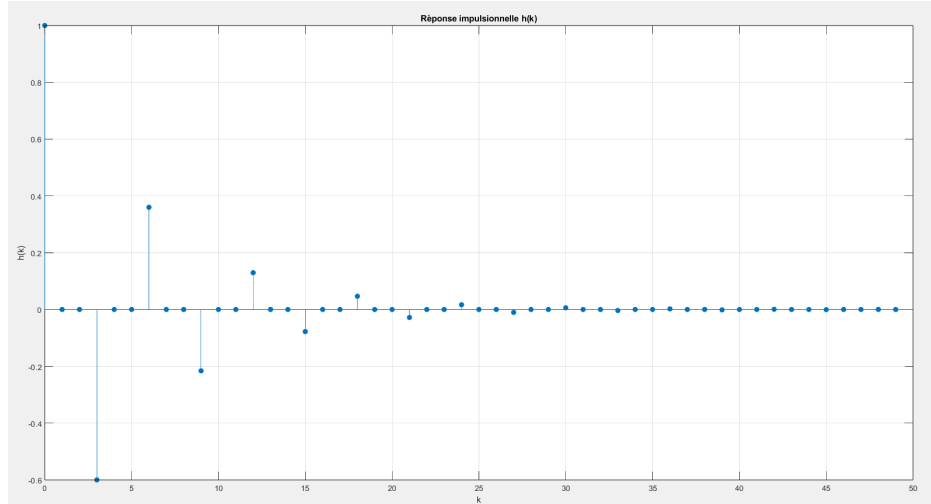


Figure 8: Réponse impulsionnelle

L'équation du filtre est :

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

On souhaite déterminer sa réponse en fréquence  $h(v)$ , avec  $v \in [-0.5, 0.5]$  la fréquence réduite (fréquence normalisée par rapport à  $F_e$ ).

En prenant la transformée en  $z$  et en évaluant sur le cercle unité  $z = e^{j2\pi v}$ , on a :

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1 + g z^{-\tau}} \Rightarrow H(v) = \frac{1}{1 + g e^{-j2\pi v \tau}}$$

On peut également réécrire cette expression sous forme factorisée :

$$h(v) = 1 + g e^{-j2\pi v \tau}$$

**Module de la réponse en fréquence :**

$$|h(v)| = |1 + g e^{-j2\pi v \tau}| = \sqrt{1 + 2g \cos(2\pi v \tau) + g^2}$$

**Phase de la réponse en fréquence :**

$$\arg(h(v)) = \tan^{-1} \left( \frac{g \sin(-2\pi v \tau)}{1 + g \cos(-2\pi v \tau)} \right)$$

**Maxima :** On remarque que le module est maximal lorsque :

$$\cos(2\pi v \tau) = -1 \Rightarrow 2\pi v \tau = (2k + 1)\pi \Rightarrow v = \frac{2k + 1}{2\tau}$$

**Valeur du module au maximum :**

$$|h(v_k)| = \sqrt{1 - 2g + g^2} = |1 - g| \quad (\text{en fait, ce minimum})$$

**Module maximal (constructif) :** lorsque  $\cos(2\pi v \tau) = 1$ , donc :

$$v = \frac{k}{\tau} \Rightarrow |h(v_k)| = |1 + g|$$

Le filtre présente une structure périodique, ce qui est caractéristique d'un **filtre en peigne**.

$$h(v) = 1 + g e^{-j2\pi v \tau} \quad \text{avec un motif périodique de pics et creux dans le spectre}$$

On souhaite tracer sur un même graphique :

- Le module de la réponse en fréquence théorique  $|h(v)| = |1 + ge^{-j2\pi v\tau}|$
- Le module obtenu numériquement via la Transformée de Fourier Discrète (DFT) de la réponse impulsionnelle

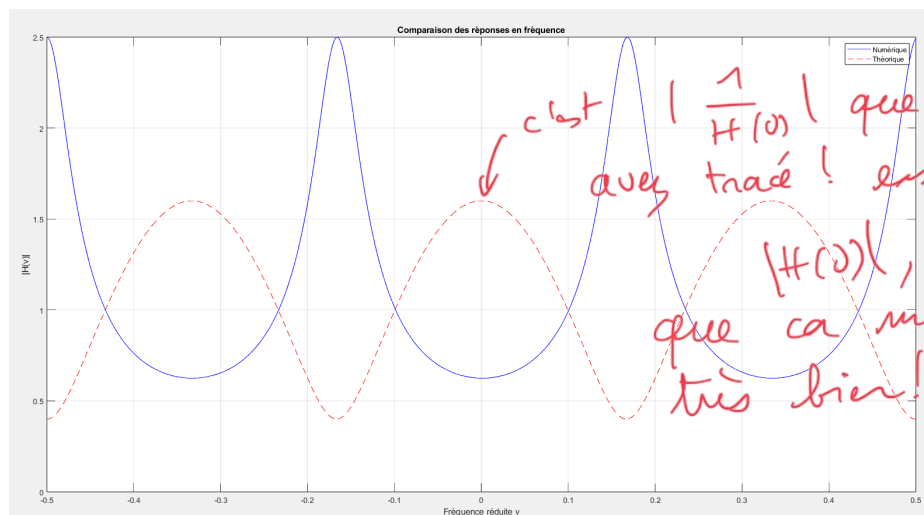


Figure 9: Réponse impulsionnelle

#### Analyse :

On observe les éléments suivants :

- Le profil de la courbe numérique (en bleu) présente des pics réguliers, caractéristiques d'un **filtre en peigne**.
- La courbe théorique (en rouge pointillée) suit la même forme générale, avec une périodicité identique.
- De légères différences d'amplitude sont visibles : les pics de la courbe numérique sont légèrement plus élevés que ceux de la courbe théorique.

#### Explication des différences :

- La réponse impulsionnelle est, en théorie, infinie. Or, pour le calcul numérique, on utilise une troncature (nombre fini d'échantillons).
- La résolution fréquentielle de la DFT est limitée par la longueur de la réponse impulsionnelle (ici,  $N$ ).
- Des effets de bord (fenêtrage implicite) peuvent apparaître à cause de cette troncature.

**Conclusion :** La comparaison est très satisfaisante et valide le modèle théorique du filtre. Les faibles écarts sont dus aux limitations inhérentes à la simulation numérique et sont acceptables dans la pratique.

#### Programmation de l'effet de delay

Après une analyse qualitative préalable du filtre, on peut chercher à l'appliquer aux signaux d'instruments fournis. On code sur matlab la fonction associée au filtre:

$$y(k) = x(k) - gy(k - \tau)$$

Ce script applique l'effet sur un fichier audio d'accord de piano avec les paramètres :

$$\tau = 0,25 \cdot F_e, \quad g = 0,9$$

ca ne va pas !  
il faut utiliser la fonction filter !

Nous avons programmé une fonction `effet_delay_filtre` qui applique l'effet de délai avec une moyenne glissante de longueur  $K$  dans la boucle de rétroaction.

La relation utilisée est :

$$y(k) = x(k) - \frac{g}{K} \sum_{n=0}^{K-1} y(k - \tau - n)$$

À l'écoute, le son filtré semble perdre progressivement ses hautes fréquences. Cette observation est confirmée par l'analyse fréquentielle : la moyenne glissante sur  $K$  échantillons dans la boucle de rétroaction agit comme un **filtre passe-bas**.

Le module de la réponse en fréquence du filtre combiné s'écrit :

$$|H(\omega)| = \left| 1 - g e^{-j\omega\tau} \cdot \frac{\sin(K\omega/2)}{K \sin(\omega/2)} \right|^{-1}$$

Le facteur

$$\frac{\sin(K\omega/2)}{K \sin(\omega/2)}$$

Avez-vous vérifié avec Matlab si c'est bon ?

se comporte comme un filtre passe-bas, s'annulant aux fréquences multiples de  $2\pi/K$ . À chaque passage dans la boucle de rétroaction, l'énergie des hautes fréquences est atténuée par un facteur  $K$ , ce qui explique l'atténuation progressive perçue comme un effet de « voile » sur le son.

Pour le justifier, on trace numériquement la réponse en fréquence du système

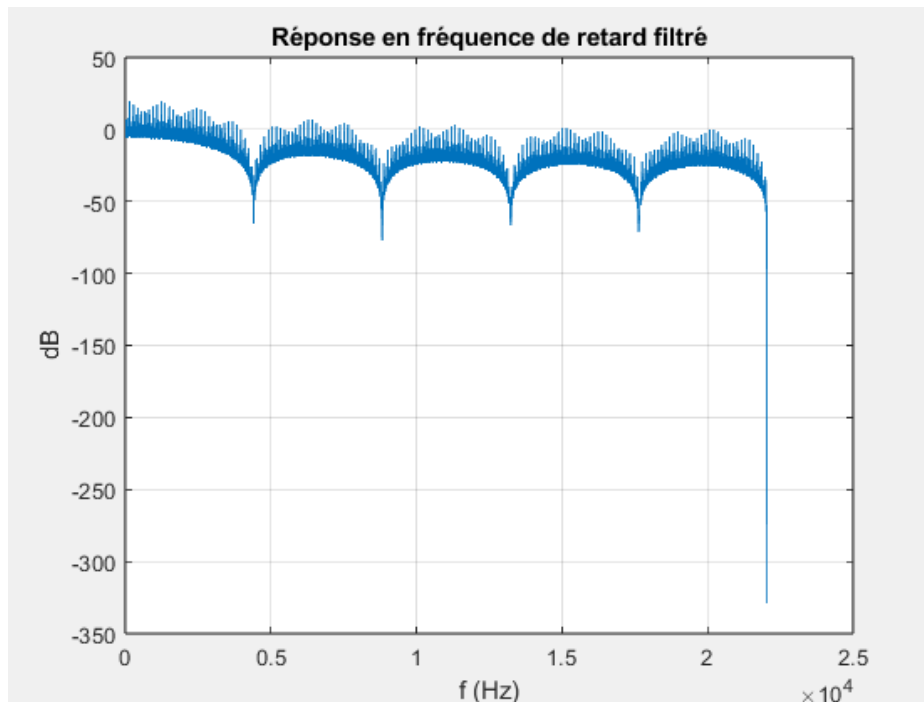


Figure 10: Module de la réponse fréquentielle

On constate que les hautes fréquences sont atténuées à cause de l'effet du filtre en moyenne glissante, ce qui donne à l'écoute une sonorité plus "mat" ou adoucie.

1. La corrélation constitue un outil fiable pour estimer la réponse impulsionnelle, en particulier lorsque le signal d'excitation est proche d'une impulsion de Dirac.
2. L'utilisation de la convolution via la transformée de Fourier rapide (FFT) permet de simuler une réverbération réelle tout en réduisant considérablement le coût de calcul, d'un facteur de 10 à 20, sans perte perceptible de qualité sonore.

3. Le délai à rétroaction met en évidence la relation entre réponse impulsionnelle et structure en peigne dans le domaine fréquentiel. L'ajout d'un filtre dans la boucle permet de modéliser l'atténuation causée par l'air ou les matériaux environnants.
4. La conception en temps réel de plug-ins audio repose sur plusieurs critères clés : la stabilité du système ( $|g| < 1$ ), la gestion de la latence (par exemple via le traitement par blocs en FFT) et le respect de la couleur spectrale souhaitée (comme une pente typique de  $-6$  dB/octave).