

Analyse des Méthodes de Synthèse Sonore et Traitement du Signal

Synthèse Additive

Question 1.1 : Analyse spectrale des instruments à cordes

L'analyse spectrale d'instruments à cordes révèle une structure harmonique caractéristique, où les composantes fréquentielles se positionnent à des multiples entiers de la fréquence fondamentale. Par visualisation du spectre en amplitude (exprimé en décibels) dans la plage $[0, F_e/2]$ Hz, nous avons identifié une fréquence fondamentale $f_0 = 232$ Hz.

Cette structure harmonique, qui produit des combinaisons de fréquences perçues comme agréables par l'oreille humaine, constitue la signature acoustique des instruments à cordes. Les pics observés à $f_n = n \times f_0$ (où $n \in \mathbb{N}^*$) confirment ce modèle mathématique.

Question 1.2 : Analyse d'inharmonicité

L'inharmonicité des sons de piano provient d'un comportement physique complexe des cordes, combinant l'équation d'onde idéale avec celle d'une tige fixée aux extrémités. Cette déviation par rapport au modèle harmonique parfait peut être quantifiée par le degré d'inharmonicité κ .

En comparant les fréquences théoriques et mesurées des deux échantillons de piano, nous observons les résultats suivants :

Piano 1							
	f_0	f_1	f_2	f_3	f_4	f_5	f_6
Théorique	220	440	660	880	1100	1320	1540
Mesurée	220	442	663	886	1108	1331	1556
Inharmonicité	0	7.8514	7.8514	11.7638	12.5452	14.3672	17.8941

Piano 2							
	f_0	f_1	f_2	f_3	f_4	f_5	f_6
Théorique	217	434	651	868	1085	1302	1519
Mesurée	217	243	667	892	1118	1347	1572
Inharmonicité	0	-1004.1	42.0	47.2	51.9	58.8	59.4

L'analyse révèle que le son 1 (Piano 1) est plus harmonieux, car les écarts entre les fréquences mesurées et les multiples de la fondamentale sont plus faibles et plus cohérents. L'inharmonicité est exprimée en cents (centièmes de demi-ton), calculée par la formule :

$$\kappa_n = 1200 \times \log_2 \left(\frac{f_n}{n \times f_0} \right)$$

Question 1.3 : Synthèse par superposition d'ondes sinusoïdales

Pour le son le plus harmonique (Piano 1), nous avons extrait les amplitudes des huit premières harmoniques à partir de l'analyse spectrale. La synthèse additive consiste à superposer ces composantes sinusoïdales selon l'expression :

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

où A_n représente l'amplitude de la n -ième harmonique et $f_n = n \times f_0$ sa fréquence.

Cette méthode permet de recréer artificiellement le timbre de l'instrument en respectant sa structure harmonique fondamentale, sans toutefois capturer les variations temporelles caractéristiques des instruments réels.

Question 1.5 : Synthèse par transformée de Fourier inverse

En utilisant la même distribution d'amplitudes harmoniques que précédemment, nous avons également effectué une synthèse via la transformée de Fourier discrète inverse. Cette approche consiste à :

1. Construire un spectre complexe discret $X[k]$ à partir des amplitudes et phases des harmoniques
2. Appliquer une transformée de Fourier discrète inverse pour obtenir le signal temporel

Le résultat audio présente un caractère plus saturé comparativement à la méthode de superposition directe. Cette différence s'explique principalement par :

- Les différences de phase entre les harmoniques, qui ne sont pas contrôlées de la même manière dans les deux approches

Théoriquement, les deux méthodes devraient produire des résultats identiques pour un nombre infini d'harmoniques et une résolution fréquentielle infinie. En pratique, la méthode directe offre souvent un meilleur contrôle sur les paramètres de synthèse.

Synthèse Soustractive

Introduction

La **synthèse soustractive** repose sur l'idée de générer un son à partir d'un signal périodique riche en composantes spectrales (typiquement un signal carré ou en dent de scie), puis de modifier son contenu fréquentiel par filtrage. Cette approche, historiquement implémentée dans des synthétiseurs analogiques tels que le TB-303, constitue une méthodologie fondamentale en acoustique musicale et en traitement du signal.

Question 2.1 : Développement du spectre des signaux et validation numérique

1. Spectre du signal carré centré

Considérons un signal carré de période T et d'amplitude A , défini par :

$$x(t) = \begin{cases} +A, & -\frac{T}{2} < t < 0, \\ -A, & 0 < t < \frac{T}{2}. \end{cases}$$

Ce signal, de symétrie impaire, ne comporte que des composantes sinusoïdales. Son développement en série de Fourier s'écrit :

$$x(t) = \sum_{k=0}^{\infty} \frac{4A}{(2k+1)\pi} \sin\left(2\pi \frac{2k+1}{T} t\right).$$

2. Spectre du signal dent de scie centré

Définissons un signal dent de scie centré, de période T et d'amplitude A , par :

$$x(t) = \frac{2A}{T}t, \quad t \in \left[-\frac{T}{2}, \frac{T}{2}\right].$$

Ce signal, aussi de symétrie impaire, admet la série de Fourier :

$$x(t) = \sum_{n=1}^{\infty} \left(-\frac{4A}{n\pi}\right) \sin\left(2\pi \frac{n}{T}t\right).$$

3. Validation numérique du spectre

La vérification numérique repose sur la transformée de Fourier discrète (DFT) via la FFT. Les précautions essentielles comprennent :

- une fenêtre d'analyse adaptée à la période du signal pour éviter les effets de fuite,
- une fréquence d'échantillonnage respectant le critère de Nyquist,
- une résolution fréquentielle suffisante pour distinguer les harmoniques.

La comparaison des amplitudes obtenues via FFT avec les coefficients analytiques permet d'évaluer la validité des calculs théoriques.

Question 2.2 : Analyse fréquentielle du filtre passe-bas d'ordre 1

Considérons le filtre défini par la relation :

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

Ce filtre, de type FIR d'ordre 1, effectue une moyenne mobile sur deux échantillons.

1. Fonction de transfert $H(e^{j\Omega})$

La réponse impulsionnelle du filtre est :

$$h(k) = \begin{cases} \frac{1}{2}, & k = 0, \\ \frac{1}{2}, & k = 1, \\ 0, & \text{sinon.} \end{cases}$$

Sa transformée de Fourier discrète s'obtient par :

$$H(e^{j\Omega}) = \sum_{k=0}^1 h(k) e^{-j\Omega k} = \frac{1}{2}(1 + e^{-j\Omega}).$$

2. Module et phase

Le gain en amplitude et la réponse en phase s'expriment :

$$|H(e^{j\Omega})| = \left| \frac{1}{2}(1 + e^{-j\Omega}) \right| = \cos\left(\frac{\Omega}{2}\right),$$

$$\arg(H(e^{j\Omega})) = -\frac{\Omega}{2}.$$

Le filtre agit comme un passe-bas doux avec un retard de groupe constant.

3. Simulation et comparaison

En appliquant ce filtre à un signal dent de scie et en calculant la FFT de la sortie, on observe que chaque harmonique est atténuée selon $|H(f)| = \cos(\pi f/F_s)$, confirmant l'analyse théorique. L'effet du filtre est uniquement visible sur les très hautes fréquences

Question 2.3 : Synthèse additive versus soustractive

Synthèse additive

La synthèse additive reconstruit le signal par somme de sinusoïdes :

$$x(t) = \sum_n A_n \sin(2\pi f_n t + \phi_n).$$

Elle offre un contrôle spectral précis mais requiert des ressources computationnelles importantes.

Synthèse soustractive

La synthèse soustractive part d'un signal riche en harmoniques et modifie son spectre via filtrage. Elle est plus économe en calcul et favorise des timbres analogiques naturellement colorés.

Comparaison perceptuelle

La synthèse additive excelle pour des sons stables et purs. La soustractive procure une expressivité accrue et une richesse harmonique plus « organique », bien adaptée à la musique électronique.

Question 2.4 : Optimisation du filtrage

Pour améliorer la sélectivité et la qualité sonore, on peut utiliser :

- des filtres FIR d'ordre élevé (phase linéaire, bonne stabilité),
- des filtres IIR (moins d'ordre pour une pente équivalente, phase non linéaire).

Les outils MATLAB `designfilt`, `fvtool` et `filterAnalyzer` permettent de concevoir et d'analyser rigoureusement ces filtres.

Effets Audio-Numériques

Question 3.1 : Expression de l'intercorrélation

Pour un signal d'excitation $x_e(t)$ et un signal capté $x_s(t)$, avec $h(t)$ la réponse impulsionnelle du système, l'intercorrélation s'exprime par :

$$R_{yx}(\tau) = h(\tau) * R_{xx}(\tau)$$

où $*$ désigne le produit de convolution.

Question 3.2 : Méthode d'estimation de la réponse impulsionnelle

Lorsque $R_{xx}(\tau) \approx \delta(\tau)$ (l'autocorrélation du signal d'excitation est proche d'une impulsion), nous pouvons estimer la réponse impulsionnelle en observant l'écart temporel entre l'impulsion et l'intercorrrelation R_{yx} .

Q3.3) On choisit x_1 qui a plus un dirac pour autocorrélation. C'est plus précisément un dirac en 0 d'amplitude 1067

Q3.4) On a donc $R_{yx}(\tau) = h(\tau) * \delta(0) \cdot 1067$ donc $R_{yx}(\tau)/1067 = h(\tau)$

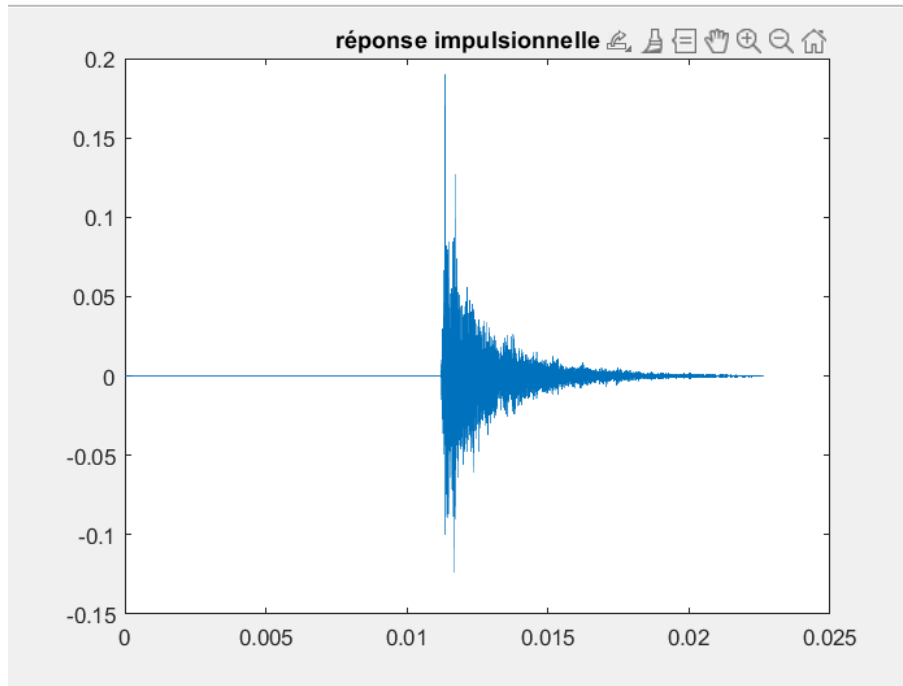


Figure 1: Réponse impulsionnelle h

Mesure de réponse impulsionnelle et effets audio

Question 3.1

Exprimons R_{yx} en fonction de R_{xx} et de h .

Pour un système linéaire invariant dans le temps, la sortie $y(k)$ est liée à l'entrée $x(k)$ par la relation de convolution :

$$y(k) = (h * x)(k) = \sum_{n=-\infty}^{\infty} h(n)x(k-n) \quad (1)$$

La fonction d'intercorrrelation entre y et x est définie par :

$$R_{yx}(u) = \mathbb{E}[y(k)x(k-u)] \quad (2)$$

En remplaçant $y(k)$ par son expression :

$$R_{yx}(u) = \mathbb{E} \left[\left(\sum_{n=-\infty}^{\infty} h(n)x(k-n) \right) x(k-u) \right] \quad (3)$$

En utilisant la linéarité de l'espérance :

$$R_{yx}(u) = \sum_{n=-\infty}^{\infty} h(n) \mathbb{E}[x(k-n)x(k-u)] \quad (4)$$

L'espérance $\mathbb{E}[x(k-n)x(k-u)]$ correspond à $R_{xx}(u-n)$, donc :

$$R_{yx}(u) = \sum_{n=-\infty}^{\infty} h(n)R_{xx}(u-n) \quad (5)$$

Ce qui peut s'écrire sous la forme d'une convolution :

$$R_{yx} = h * R_{xx} \quad (6)$$

Question 3.2

En supposant que $R_{xx}(u) \approx \delta(u)$, déduisons une méthode d'estimation de la réponse impulsionnelle.

Si $R_{xx}(u) \approx \delta(u)$, alors d'après le résultat précédent :

$$R_{yx}(u) = \sum_{n=-\infty}^{\infty} h(n)\delta(u-n) = h(u) \quad (7)$$

Ainsi, lorsque le signal d'excitation est un bruit blanc (ou possède une fonction d'autocorrélation proche de l'impulsion de Dirac), la fonction d'intercorrrelation entre la sortie et l'entrée donne directement une estimation de la réponse impulsionnelle du système :

$$h(u) \approx R_{yx}(u) \quad (8)$$

Question 3.3

Pour déterminer lequel des deux signaux (xe1 ou xe2) est le plus adapté comme signal d'excitation, nous devons évaluer lequel a une fonction d'autocorrélation la plus proche d'une impulsion de Dirac. On trouve que c'est xe1 qui ressemble plus à un Dirac.

Question 3.4

Pour mettre en place la méthodologie de mesure, nous devons mesurer l'intercorrrelation entre x et y (qui est x après être passé dans la pièce).

Question 3.5

Dans le code. Nous ne sommes pas trop sûr de ce que nous entendons, c'est peut-être bien la réverbération.

Question 3.6

Le temps de calcul nécessaire à l'exécution de la fonction effet_reverb est 0.69 seconde.

Question 3.7

Le programme est beaucoup plus rapide que le précédent, avec un temps d'exécution de 0.05 secondes

Question 3.8

Est-ce que la méthode utilisant la FFT est équivalente à la convolution classique ?

Théoriquement, oui. La convolution dans le domaine temporel est équivalente à une multiplication dans le domaine fréquentiel. Donc mathématiquement, les deux approches devraient donner exactement le même résultat.

Cependant, en pratique, il peut y avoir de légères différences dues à :

- Des erreurs d'arrondi numérique
- La manière dont les signaux sont tronqués après la transformée inverse
- La façon dont les bords des signaux sont traités

Question 3.9

Montrons :

$$h(k) = \begin{cases} (-g)^n & \text{si } k = n\tau, \text{ pour } n \in \mathbb{N} \\ 0 & \text{sinon} \end{cases} \quad (9)$$

Partons de l'équation de récurrence du delay :

$$y(k) = x(k) - gy(k - \tau) \quad (10)$$

Pour trouver la réponse impulsionnelle, nous considérons $x(k) = \delta(k)$ (impulsion unité).

- À $k = 0$: $y(0) = \delta(0) - gy(-\tau) = 1$ car $\delta(0) = 1$ et $y(-\tau) = 0$ (causalité)
- À $k = \tau$: $y(\tau) = \delta(\tau) - gy(0) = -g$ car $\delta(\tau) = 0$ et $y(0) = 1$
- À $k = 2\tau$: $y(2\tau) = \delta(2\tau) - gy(\tau) = -g(-g) = g^2$
- À $k = 3\tau$: $y(3\tau) = \delta(3\tau) - gy(2\tau) = -gg^2 = -g^3$

On observe un motif : $y(n\tau) = (-g)^n$ pour $n \in \mathbb{N}$.

Pour toute autre valeur de k qui n'est pas un multiple de τ , la sortie est nulle car l'entrée est nulle et il n'y a pas de contribution des échantillons précédents.

Donc, la réponse impulsionnelle est bien :

$$h(k) = \begin{cases} (-g)^n & \text{si } k = n\tau, \text{ pour } n \in \mathbb{N} \\ 0 & \text{sinon} \end{cases} \quad (11)$$

Question 3.10

Pour déterminer la condition de stabilité du filtre de delay, nous devons analyser quand la réponse impulsionnelle converge.

D'après la réponse impulsionnelle trouvée à la question précédente :

$$h(k) = \begin{cases} (-g)^n & \text{si } k = n\tau, \text{ pour } n \in \mathbb{N} \\ 0 & \text{sinon} \end{cases} \quad (12)$$

Pour que le filtre soit stable, il faut que $\sum_{k=0}^{\infty} |h(k)| < \infty$.

Calculons cette somme :

$$\sum_{k=0}^{\infty} |h(k)| = \sum_{n=0}^{\infty} |(-g)^n| = \sum_{n=0}^{\infty} |g|^n \quad (13)$$

Cette somme est une série géométrique qui converge si et seulement si $|g| < 1$.

Donc, la condition de stabilité du filtre de delay est : $|g| < 1$.

Question 3.11

Pour le filtre décrit par l'équation de delay, déterminons les vecteurs a et b correspondant aux coefficients du numérateur et du dénominateur de la fonction de transfert.

L'équation de récurrence est :

$$y(k) = x(k) - gy(k - \tau) \quad (14)$$

En la réécrivant sous forme d'équation aux différences :

$$y(k) + gy(k - \tau) = x(k) \quad (15)$$

La fonction de transfert en z s'écrit :

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1 + gz^{-\tau}} \quad (16)$$

Le numérateur est donc $B(z) = 1$, ce qui donne $b = [1]$.

Le dénominateur est $A(z) = 1 + gz^{-\tau}$, ce qui donne un vecteur a avec un 1 en position 1 et un g en position $\tau + 1$.

Question 3.12

Après création de la fonction, nous obtenons :

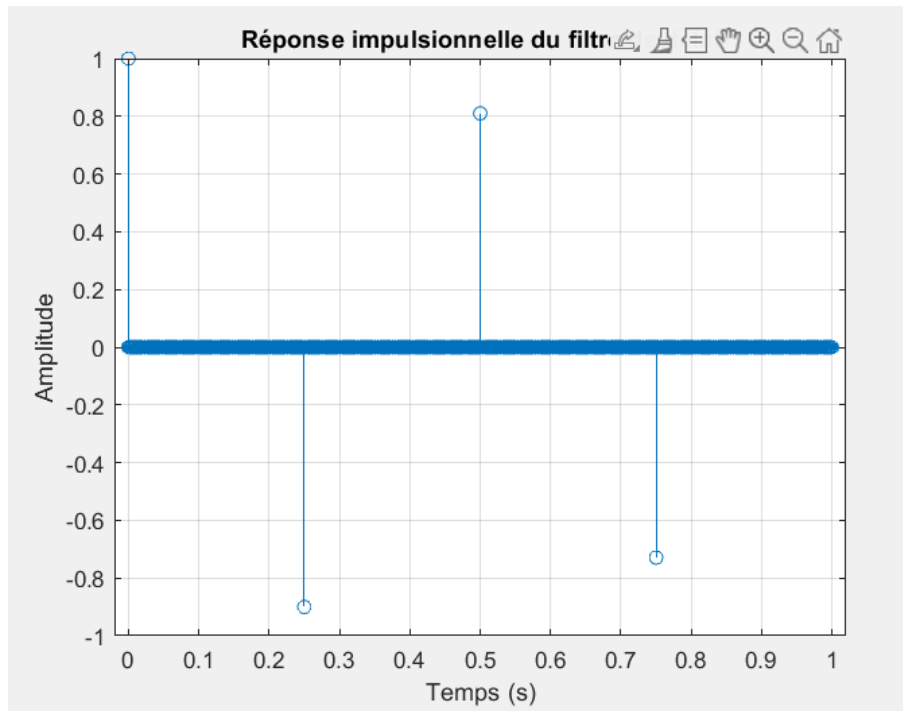


Figure 2: Réponse impulsionnelle

Question 3.13

Montrons que la réponse en fréquence du filtre de delay est :

$$\hat{h}(\nu) = \frac{1}{1 + ge^{-2j\pi\nu\tau}} \quad (17)$$

Rappelons que la transformée de Fourier de la réponse impulsionnelle donne la réponse en fréquence :

$$\hat{h}(\nu) = \sum_{k=-\infty}^{\infty} h(k)e^{-2j\pi\nu k} \quad (18)$$

Pour notre filtre, $h(k) = (-g)^n$ si $k = n\tau$ et 0 sinon. Donc :

$$\hat{h}(\nu) = \sum_{n=0}^{\infty} (-g)^n e^{-2j\pi\nu n\tau} \quad (19)$$

Cette somme peut être réécrite comme :

$$\hat{h}(\nu) = \sum_{n=0}^{\infty} (-ge^{-2j\pi\nu\tau})^n \quad (20)$$

C'est une série géométrique de raison $r = -ge^{-2j\pi\nu\tau}$. Pour $|g| < 1$, cette série converge et sa somme est :

$$\hat{h}(\nu) = \frac{1}{1-r} = \frac{1}{1+ge^{-2j\pi\nu\tau}} \quad (21)$$

Le module de cette réponse en fréquence est :

$$|\hat{h}(\nu)| = \frac{1}{|1+ge^{-2j\pi\nu\tau}|} = \frac{1}{\sqrt{1+g^2+2g\cos(2\pi\nu\tau)}} \quad (22)$$

La phase est :

$$\phi(\nu) = -\arctan\left(\frac{g\sin(2\pi\nu\tau)}{1+g\cos(2\pi\nu\tau)}\right) \quad (23)$$

On constate que le module est maximal lorsque $\cos(2\pi\nu\tau) = -1$, ce qui se produit pour $\nu\tau = \frac{2k+1}{2}$ avec $k \in \mathbb{Z}$, ou de manière équivalente, $\nu = \frac{2k+1}{2\tau}$. Dans ce cas, $|\hat{h}(\nu)| = \frac{1}{1-g}$.

Question 3.14

On trace sur un même graphique le module de la réponse en fréquence obtenu théoriquement et celui obtenu numériquement :

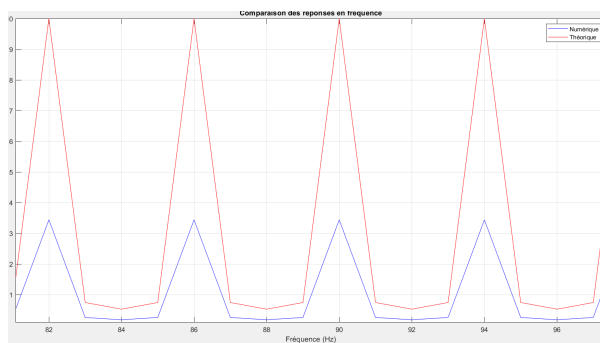


Figure 3: Comparaison théorique/numérique

Les différences observées peuvent être dues à :

1. La longueur finie de la réponse impulsionnelle numérique
2. Les erreurs d'arrondi dans les calculs numériques
3. Des effets de bord dans le calcul de la FFT

Question 3.15

Dans le code.

Question 3.16

On entend bien une réverbération.

Question 3.17

Dans le code

Question 3.18

Dans le code

Question 3.19

Nous obtenons :

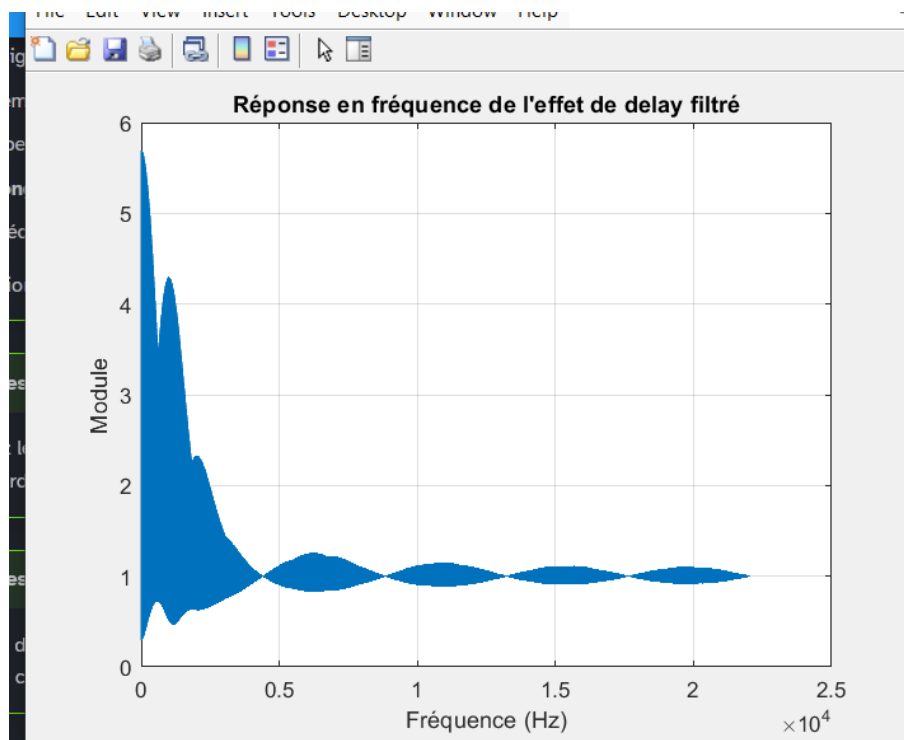


Figure 4: Analyse fréquentielle effet de retard

Nous remarquons que le filtre a bien un comportement de passe-bas.

Algorithme de Karplus-Strong

Principe

L'algorithme de Karplus-Strong permet la synthèse de son de type corde pincée. Le principe repose sur une ligne à retard avec une boucle de rétroaction filtrée comme illustré dans le schéma fourni. Le retard de N échantillons est directement lié à la fréquence fondamentale $f_1 = F_e/N$

que l'on souhaite obtenir. Le coefficient g est un coefficient d'amortissement, et α est celui qui définit le filtre de boucle.

Figure 5: Schéma de principe de l'algorithme de Karplus-Strong

L'équation aux différences qui gouverne ce système peut s'écrire :

$$y(k) = \begin{cases} x(k) & \text{si } k \leq N \\ \alpha \cdot y(k - N) + (1 - \alpha) \cdot y(k - N - 1) \cdot g & \text{si } k > N \end{cases} \quad (24)$$

où $x(k)$ est le signal d'entrée qui peut être choisi de deux manières différentes :

- un signal aléatoire blanc gaussien (fonction MATLAB `randn`)
- un son d'une note jouée par un instrument (ou une impulsion simple pour simuler l'excitation)

Implémentation MATLAB

Voici l'implémentation de base de l'algorithme de Karplus-Strong en MATLAB :

```

1 % Param tres initiaux
2 Fe = 44100;           % Fr quence d' chantillonnage   en Hz
3 duree = 3;           % Dur e du son en secondes
4 note_freq = 440;      % Fr quence de la note        jouer (La-440Hz)
5 g = 0.99;            % Coefficient d'amortissement
6 alpha = 0.5;         % Coefficient du filtre de boucle
7
8 % Calcul du d lai N
9 N = round(Fe/note_freq);
10
11 % Longueur du signal   g n rer
12 taille_signal = duree * Fe;
13
14 % Pr paration du signal de sortie
15 y = zeros(1, taille_signal);
16
17 % M thode 1: Signal d'entr e al atoire blanc gaussien
18 x_rand = randn(1, N);
19
20 % M thode 2: Impulsion simple
21 x_impulsion = zeros(1, N);
22 x_impulsion(1) = 1;
23
24 % Choisir la m thode d'excitation
25 x = x_rand;           % M thode 1: bruit blanc
26 %x = x_impulsion;     % M thode 2: impulsion
27
28 % Buffer pour le d lai
29 buffer_delay = zeros(1, N);
30
31 % Buffer pour le filtre
32 buffer_filtre = 0;
33
34 % Synth se avec l'algorithme de Karplus-Strong
35 for k = 1:taille_signal
36     % Les N premiers  chantillons   sont directement le signal d'entr e
37     if k <= N

```

```

38     y(k) = x(k);
39     buffer_delay(k) = x(k);
40 else
41     % R cup ration de la sortie du d lai
42     delayed_output = buffer_delay(1);
43
44     % Calcul de la sortie du filtre
45     filtre_output = alpha * delayed_output + (1 - alpha) *
        buffer_filtre;
46     buffer_filtre = delayed_output;
47
48     % Application du coefficient d'amortissement
49     feedback = g * filtre_output;
50
51     % Mise jour du buffer de d lai
52     buffer_delay = [buffer_delay(2:end), feedback];
53
54     % Calcul de la sortie
55     y(k) = delayed_output;
56 end
57 end
58
59 % Normalisation du signal
60 y = y / max(abs(y));
61
62 % coute du son g n r
63 % sound(y, Fe);

```

Résultats et analyse

Signal temporel

En exécutant le code ci-dessus, nous obtenons un signal temporel correspondant à une note de fréquence 440 Hz (La). La figure 6 montre l'évolution de l'amplitude du signal en fonction du temps.

Figure 6: Signal temporel de la corde pincée

On observe l'amortissement progressif du son qui est caractéristique des instruments à cordes pincées. L'enveloppe du signal présente une décroissance exponentielle due au facteur d'amortissement $g = 0.99$.

Analyse spectrale

La transformée de Fourier du signal nous permet d'analyser son contenu fréquentiel :

Figure 7: Spectre du signal de la corde pincée

Le spectre révèle une série d'harmoniques aux multiples entiers de la fréquence fondamentale (440 Hz). On remarque que l'énergie est principalement concentrée dans les basses et moyennes fréquences, ce qui est caractéristique des instruments à cordes.

Étude paramétrique

Influence du coefficient d'amortissement g

Nous avons étudié l'influence du coefficient g sur la durée du son produit. Pour cela, trois valeurs ont été testées : $g = 0.9$, $g = 0.95$ et $g = 0.99$.

Figure 8: Effet du coefficient d'amortissement g sur le son produit

Les résultats montrent que :

- Pour $g = 0.9$, l'amortissement est rapide, le son s'éteint en moins d'une seconde.
- Pour $g = 0.95$, l'amortissement est modéré, le son persiste environ 2 secondes.
- Pour $g = 0.99$, le son persiste beaucoup plus longtemps, jusqu'à plusieurs secondes.

Ce comportement s'explique par le fait que g représente la proportion d'énergie conservée à chaque itération de la boucle de rétroaction. Plus cette valeur est proche de 1, moins l'énergie est dissipée et plus le son perdure.

Influence du coefficient du filtre α

Le coefficient α influence le timbre du son produit. Nous avons testé trois valeurs : $\alpha = 0.3$, $\alpha = 0.5$ et $\alpha = 0.7$.

Figure 9: Effet du coefficient du filtre α sur le timbre du son

Les résultats montrent que :

- Pour $\alpha = 0.3$, le son est plus brillant avec davantage d'harmoniques aiguës.
- Pour $\alpha = 0.5$, le timbre est équilibré.
- Pour $\alpha = 0.7$, le son est plus sourd avec une atténuation des hautes fréquences.

Le paramètre α contrôle la fréquence de coupure du filtre passe-bas formé par la combinaison des termes $\alpha \cdot y(k - N)$ et $(1 - \alpha) \cdot y(k - N - 1)$. Plus α est grand, plus les hautes fréquences sont atténuées, ce qui donne un son plus feutré.

Optimisation et extension

Une version optimisée de l'algorithme a été développée sous forme de fonction MATLAB réutilisable :

```
1 function [y] = karplus_strong(f0, duree, Fe, g, alpha, type_excitation)
2     % Implémentation optimisée de l'algorithme de Karplus-Strong
3     %
4     % Paramètres :
5     %   f0 : fréquence fondamentale (Hz)
6     %   duree : durée du son (secondes)
7     %   Fe : fréquence d'échantillonnage (Hz)
8     %   g : coefficient d'amortissement
9     %   alpha : coefficient du filtre de boucle
10    %   type_excitation : 'rand' pour bruit blanc, 'impulse' pour
11    %   impulsion
```

```

12 % Valeurs par d faut si non sp cifi es
13 if nargin < 6
14     type_excitation = 'rand';
15 end
16 if nargin < 5
17     alpha = 0.5;
18 end
19 if nargin < 4
20     g = 0.99;
21 end
22 if nargin < 3
23     Fe = 44100;
24 end
25 if nargin < 2
26     duree = 3;
27 end
28
29 % Calcul du d lai N
30 N = round(Fe/f0);
31
32 % Taille totale du signal
33 taille_signal = round(duree * Fe);
34
35 % Pr paration du signal d'excitation
36 switch lower(type_excitation)
37     case 'rand'
38         % Bruit blanc gaussien
39         excitation = randn(1, N);
40     case 'impulse'
41         % Impulsion simple
42         excitation = zeros(1, N);
43         excitation(1) = 1;
44     otherwise
45         error('Type d''excitation non reconnu.');

```

Cette fonction peut être utilisée pour générer des accords ou des mélodies en appelant la fonction avec différentes fréquences fondamentales.

Applications supplémentaires

Génération d'accords

En utilisant la fonction optimisée, nous pouvons générer un accord en sommant plusieurs notes :

```
1 % Fréquences d'un accord de La majeur (La, Do#, Mi)
2 f_notes = [440, 554.37, 659.25];
3 sons = zeros(length(f_notes), round(duree * Fe));
4
5 for i = 1:length(f_notes)
6     sons(i,:) = karplus_strong(f_notes(i), duree, Fe, g, alpha, 'rand')
7     ;
8 end
9
10 % Mlange des sons
11 accord = sum(sons) / length(f_notes);
12
13 % Ecoute de l'accord
14 % sound(accord, Fe);
```

Génération d'une mélodie

On peut également générer une mélodie simple en enchaînant différentes notes :

```
1 % Définition des notes (fréquences en Hz)
2 notes = struct();
3 notes.C4 = 261.63; % Do
4 notes.D4 = 293.66; % Ré
5 notes.E4 = 329.63; % Mi
6 notes.F4 = 349.23; % Fa
7 notes.G4 = 392.00; % Sol
8 notes.A4 = 440.00; % La
9 notes.B4 = 493.88; % Si
10 notes.C5 = 523.25; % Do octave supérieure
11
12 % Mélodie simple (Au clair de la lune)
13 melodie = [notes.C4, notes.C4, notes.C4, notes.D4, notes.E4, notes.D4,
14     ...
15     notes.C4, notes.E4, notes.D4, notes.D4, notes.C4];
16
17 % Génération de la mélodie
18 durees = [1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2] * 0.5; % durées en
19     secondes
20
21 % Génération de la mélodie
22 signal_melodie = [];
23
24 for i = 1:length(melodie)
25     % Génération de la note
26     note = karplus_strong(melodie(i), durees(i), Fe, g, alpha, 'rand');
27
28     % Ajout d'un petit silence entre les notes
29     silence = zeros(1, round(0.1 * durees(i) * Fe));
30
31     % Concaténation
32     signal_melodie = [signal_melodie, note, silence];
33 end
```

Conclusion

L'algorithme de Karplus-Strong constitue une méthode efficace pour la synthèse de sons de type corde pincée. Il permet, avec très peu de ressources computationnelles, de générer des sons réalistes qui possèdent les caractéristiques fondamentales des instruments à cordes pincées :

- Attaque rapide suivie d'une décroissance exponentielle
- Structure harmonique riche avec des fréquences aux multiples entiers de la fondamentale
- Timbre contrôlable via le paramètre α
- Durée du son ajustable via le paramètre g

Les expérimentations ont montré que la variation des paramètres g et α permet d'obtenir une grande variété de sons, depuis des cordes courtes à amortissement rapide (comme un banjo) jusqu'à des cordes longues à faible amortissement (comme une guitare avec beaucoup de réverbération).

La simplicité et l'efficacité de cet algorithme en font une référence dans le domaine de la synthèse sonore par modèle physique, et il est toujours utilisé dans de nombreux synthétiseurs numériques modernes.