Traitement des Signaux Aléatoires : Prédiction linéaire d'un signal par modélisation Auto-Régressive Application au débruitage

4 ETI – CPE Lyon

Travaux Pratiques TSA

Noms, Prénoms : FRANCOIS Axel, PINCEMIN Alexis

Groupe : D

Date : 26/10/21

Objectif – Débruitage d'un enregistrement sonore par prédiction linéaire basée sur un modèle auto-régressif.

L'ensemble des fonctions Matlab nécessaires est à récupérer sur la plateforme *CPe-campus* sous l'archive `Fichiers_TP_AR.zip`.

I Préparation : Modélisation auto-régressive d'un signal aléatoire

Soient $(s_n,\, n = 1, \ldots, N)$, les échantillons d'un signal aléatoire réel, stationnaire. On se propose de trouver **un modèle** tel que l'on puisse prédire linéairement la valeur de s_n à partir des échantillons précédents de s

$$\hat{s}_n = \sum_{k=1}^{\infty} h[k]\, s_{n-k}. \tag{1}$$

Il s'agit donc de déterminer les coefficients $h[k]$, $k = 1, 2, \ldots$ minimisant l'erreur de prédiction :

$$\varepsilon_n = s_n - \hat{s}_n = s_n - \sum_{k=1}^{\infty} h[k] s_{n-k} \tag{2}$$

au sens de l'erreur quadratique moyenne (i.e. puissance moyenne minimale) $P_\varepsilon = \mathbb{E}\{\varepsilon_n^2\} = \sigma^2$

I.1

Quel principe de construction permet de garantir une erreur quadratique moyenne minimale ?

réponse

Le principe de construction qui permet de garantir une erreur quadratique moyenne minimale est le principe d'ortogonalité. □

I.2

Par application de ce principe, montrer que l'erreur de prédiction ε_n est un bruit blanc.

réponse

Par principe d'otrogonalité on a $\mathbb{E}\{\varepsilon_n s_{n-k}\} = 0 \ \forall k > 0$

$\gamma_\varepsilon(m) = \mathbb{E}\{\varepsilon_n \varepsilon_{n-m}\} = \mathbb{E}\{\varepsilon_n(s_n - \hat{s}_n)\} = \mathbb{E}\{\varepsilon_n(s_n - \sum_{j=1}^{\infty} h[j] s_{n-m-j}\} = \mathbb{E}\{\varepsilon_n s_{n-m}\} - \mathbb{E}\{\varepsilon_n \sum_{j=1}^{\infty} h[j] s_{n-m-j}\} =$

$\mathbb{E}\{\varepsilon_n s_{n-m}\} - \sum_{j=1}^{\infty} h[j] \mathbb{E}\{\varepsilon_n s_{n-m-j}\} = 0 - \sum_{j=1}^{\infty} h[j] \times 0 = 0$ □

I.3

En pratique, il faut limiter l'ordre du modèle à $M < \infty$. Dans ces conditions et toujours par application de ce même principe de construction, établir le système d'équations linéaires, dont les coefficients $\{h[k],\ k = 1, \dots M\}$ sont les solutions.

réponse

Par principe d'otrogonalité on a $\mathbb{E}\{\varepsilon_n s_{n-m}\} = 0 \ \forall m > 0$

$\mathbb{E}\{(s_n - \hat{s}_n) s_{n-m}\} = \mathbb{E}\{s_n s_{n-m}\} - \mathbb{E}\{\hat{s}_n s_{n-m}\} = \gamma_s(m) - \sum_{k=1}^{M} h[k] \mathbb{E}\{s_{n-k} s_{n-m}\} = \gamma_s(m) - \sum_{k=1}^{M} h[k] \gamma_s(m-k) = 0$

On a donc $\gamma_s(m) = \sum_{k=1}^{M} h[k] \gamma_s(m-k)$

$$\begin{pmatrix} \gamma_s(1) & \gamma_s(0) & \gamma_s(1) & \dots & \gamma_s(M-1) \\ \gamma_s(2) & \gamma_s(1) & \gamma_s(0) & \dots & \gamma_s(M-2) \\ . & . & . & & . \\ . & . & . & & . \\ . & . & . & & . \\ \gamma_s(M) & \gamma_s(M-1) & \gamma_s(M-2) & \dots & \gamma_s(0) \end{pmatrix} \begin{pmatrix} -1 \\ h[1] \\ . \\ . \\ . \\ h[M] \end{pmatrix} = \begin{pmatrix} -\sigma_\varepsilon^2 \\ 0 \\ . \\ . \\ . \\ 0 \end{pmatrix}$$ □

I.4

Calculer la puissance de l'erreur de prédiction P_ε en fonction de l'autocorrélation $\gamma_s(k)$ du signal et des coefficients $\{h[k],\ k = 1, \dots M\}$.

réponse

$P_\varepsilon = \sigma_\varepsilon^2 = \mathbb{E}\{(s_n - \hat{s}_n)(s_n - \hat{s}_n)\} = \mathbb{E}\{(s_n - \hat{s}_n) s_n\} - \mathbb{E}\{(s_n - \hat{s}_n)\hat{s}_n\} = \gamma_s[0] - \sum_{k=1}^{M} h[k] \gamma_s[k]$ □

I.5

Insérer cette relation dans le système d'équations linéaires obtenu à la question I.3.

réponse

$$\begin{pmatrix} \gamma_s(0) & \gamma_s(1) & \gamma_s(2) & ... & \gamma_s(M) \\ \gamma_s(1) & \gamma_s(0) & \gamma_s(1) & ... & \gamma_s(M-1) \\ \gamma_s(2) & \gamma_s(1) & \gamma_s(0) & ... & \gamma_s(M-2) \\ . & . & . & & . \\ . & . & . & & . \\ . & . & . & & . \\ \gamma_s(M) & \gamma_s(M-1) & \gamma_s(M-2) & ... & \gamma_s(0) \end{pmatrix} \begin{pmatrix} \phi(0) \\ \phi(1) \\ \phi(2) \\ . \\ . \\ . \\ \phi(M) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ . \\ . \\ . \\ 0 \end{pmatrix}$$

Avec $\phi(0) = \frac{1}{\sigma_\varepsilon^2}$ et $\phi(k) = \frac{-h[k]}{\sigma_\varepsilon^2}$ □

II Manipulation

II.1 Signal et contexte

Charger le signal audio `ProtestMonoBruit.wav` avec la commande :

`[s,Fs]= audioread('ProtestMonoBruit.wav') ;`

où s et Fs correspondent respectivement au signal échantillonné et à la fréquence d'échantillonnage.
En indiquant le code correspondant, afficher le signal (axe temporel gradué en secondes). Les pics aléatoires superposés au signal sont une affection classique des sons numérisés à partir de disques vinyles, qui se manifestent par un *bruit de craquement* lorsqu'on écoute le fichier audio :

`sound(s,Fs) ;`

code

```
%% 1 Signal et contexte

[s,Fs]= audioread('ProtestMonoBruit.wav');
t= 0:1/Fs:(length(s)-1)*1/Fs;

figure()
plot(t,s)
title('Signal sonore')
xlabel('temps (en secondes)')
ylabel('Amplitude')
```

□

figures

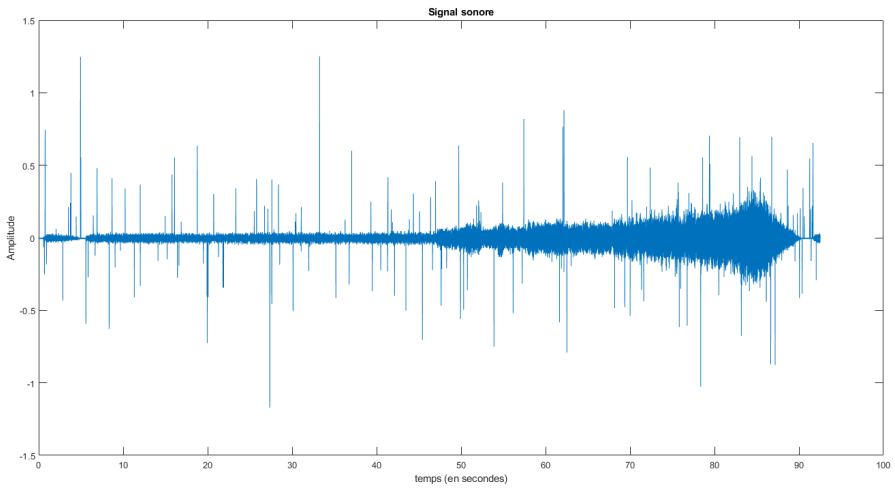

FIGURE 1 – Signal sonore échantillonné (*en secondes*)

□

On se propose de restaurer le signal initial en *débruitant* l'enregistrement.
Quelle(s) solution(s) simple(s) pourrait on envisager pour supprimer (ou atténuer) l'effet de ces pics ? En quoi ne seraient elles pas satisfaisantes ?

réponse

On pourrais utiliser un filtre passe bas. En effet, une perturbation aléatoire en pic correspond en Fourier à un signal très haute fréquence. Cette solution n'est cependant pas la plus adaptée : si les perturbations sont trop proches les unes des autres, ce qui est possible puisque les perturbations sont aléatoires, on risque de devoir filtrer avec une fréquence de coupure qui rend inutilisable le signal final. L'autre solution serait de filtrer les pics avec une amplitudes trop forte, mais cela ne retirerais pas toutes les perturbations. □

La solution retenue ici, consiste à modéliser l'enregistrement audio (qui n'est pas un bruit blanc !) par un **processus auto-régressif d'ordre M** ($AR(M)$). Ce modèle est ensuite utilisé pour prédire l'échantillon s_n du signal à partir des M échantillons précédents $\{s_{n-m},\, 1 \leq m \leq M)\}$. Puisque l'apparition des pics de craquement est **aléatoire** et supposée **indépendante** du signal, le modèle $AR(M)$ n'aura pas le pouvoir de les prédire.

II.2 Estimation de la fonction d'autocorrélation.

Pour limiter les temps de calcul, on limitera l'analyse du signal à l'intervalle $t \in [60, 70]$ secondes.
Sur ce segment, estimer la fonction d'autocorrélation $\gamma_s[k] = \gamma_s(kTs)$, pour $-K \leq k \leq K$. Pour cela, on utilisera l'instruction suivante : `[R,lags] = xcorr(x,K,'biased')`, où R est le vecteur des corrélations et `lags`, le vecteur des retards $-K \leq k \leq K$. Afficher le code correspondant et le résultat pour $K = 200$ (on n'hésitera pas à faire un zoom autour de la zone d'intérêt...)

code

```
%% 2 Estimation de la fonction d autocorrelation

%resitriction [60,70]
indexmin = find(t<=60);
indexmax = find(t>=70);
s1 = s(indexmin(end):indexmax(1));
t1 = t(indexmin(end):indexmax(1));

figure()
plot(t1,s1)

%calcul autocorrelation
K = 200;
[R,lags] = xcorr(s1,K,'biased');
figure()
stem(lags/Fs,R)
title('Estimation de la fonction d''autocorrelation')
xlabel('temps (en secondes)')
ylabel('Amplitude')
```

□

figures

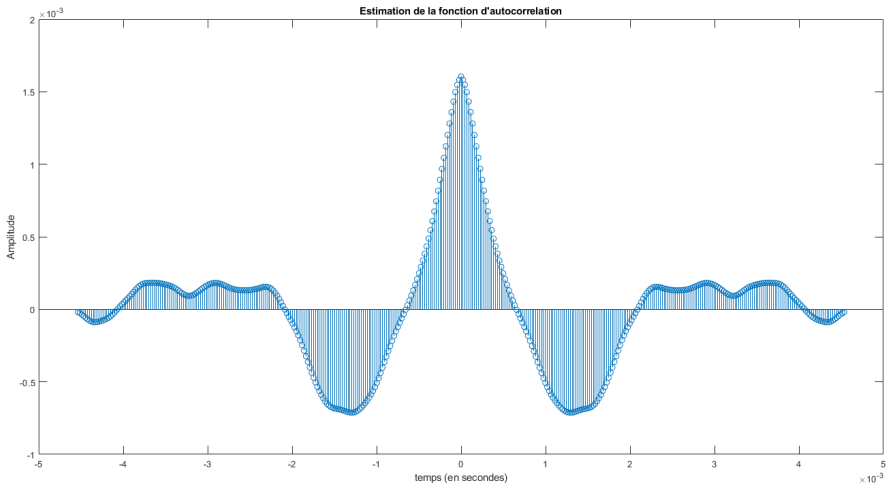

FIGURE 2 – Estimation de la fonction d'autocorrélation $\gamma_s[k] = \gamma_s(kTs)$ pour $-200 \leq k \leq 200$

□

A partir de cette fonction, justifier le choix $M \approx 20$ pour l'ordre du modèle $AR(M)$.

réponse

On choisi $M \approx 20$ car on prend en compte le rayon de corrélation qui se situe à K = 29 donc après celui-ci les échantillons sont considérés comme non corrélés. □

II.3 Identification du modèle $AR(M)$

En fixant alors l'ordre du modèle auto-régressif à $M = 20$, programmer et résoudre l'équation matricielle (complète) obtenue à la question I.5 de la préparation. Utiliser pour cela, la fonction `toeplitz.m` de Matlab.

Note : γ_s est la fonction d'autocorrélation **empirique** estimée à partir des échantillons de s. La matrice de Toeplitz de la question I.5 de la préparation peut donc ne pas être inversible. On utilisera alors la commande `pinv.m` de Matlab pour calculer la matrice pseudo-inverse (de Moore-Penrose) qui elle, existe toujours.

Reproduisez ci-dessous, le code développé pour le calcul des coefficients.

code

```
%% 3 Identification du modele AR(M)
M = 20;
MGamma = toeplitz(R(K+1 : K + M+1));
Sol = zeros(M+1,1);
Sol(1,1) = 1;
Phi = pinv(MGamma) * Sol;
H = -Phi(2:M+1) / Phi(1);
figure()
plot(1:1:M , H')
title('coefficients du modele AR(20)')
xlabel('Indices des coefficients')
ylabel('Amplitude')
```

□

Afficher les coefficients du modèle $AR(M)$ $\{h[k],\ k = 1, \dots M\}$ ainsi obtenus.

figures

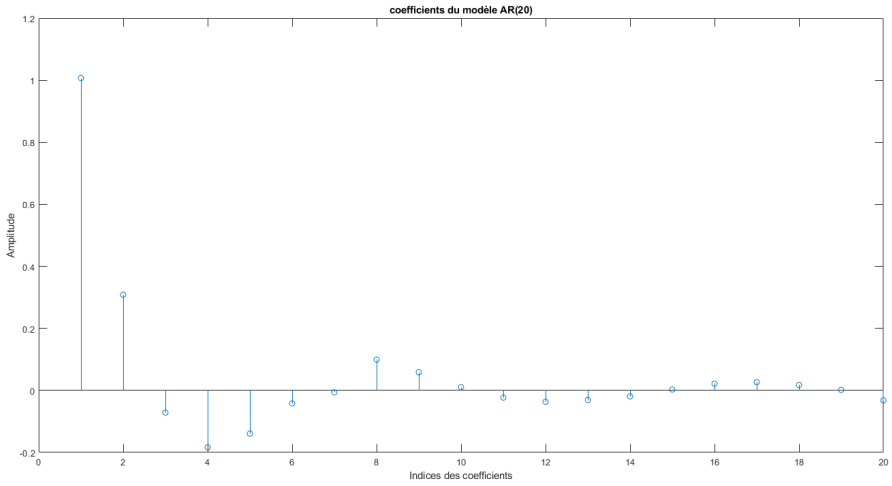

FIGURE 3 – Coefficients du modèle $AR(20)$ $\{h[k],\ k = 1, \dots 20\}$

□

II.4 Prédiction linéaire

En utilisant les coefficients $(h[k],\ k = 1, \dots, M)$ du modèle $AR(20)$ ainsi identifié, calculer la prédiction à un pas de temps $\hat{s}_n$ de s_n à partir des échantillons précédents $(s_{n-k},\ k = 1, \dots, M)$, pour $n \geq M + 1$. Reproduisez ci-dessous, le code développé pour effectuer la prédiction et le calcul de l'erreur de prédiction.

code

```
%% 4 Prediction lineaire
Schapeau = conv(s1, H, 'valid');
figure()
subplot(2,1,1)
hold on
plot(t1,s1)
plot(t1(M+1:end),Schapeau(1:end-1))
title('Prediction du signal superposee au signal original')
xlabel('temps (en secondes)')
ylabel('Amplitude')
legend('s','$\hat{s}$','interpreter','latex')
subplot(2,1,2)
epsilon = s1(M:end)-Schapeau;
plot(t1(M:end),abs(epsilon))
title('Valeur absolue de l erreur de prediction')
xlabel('temps (en secondes)')
ylabel('Amplitude')
```

□

Afficher (en `subplot(211)`) la prédiction du signal ainsi obtenue superposée au signal original et faites un zoom sur un craquement de votre choix.
Sur la même figure (en `subplot(212)`) afficher la valeur absolue de l'erreur de prédiction $\varepsilon_n = \hat{s}_n - s_n$.

figures

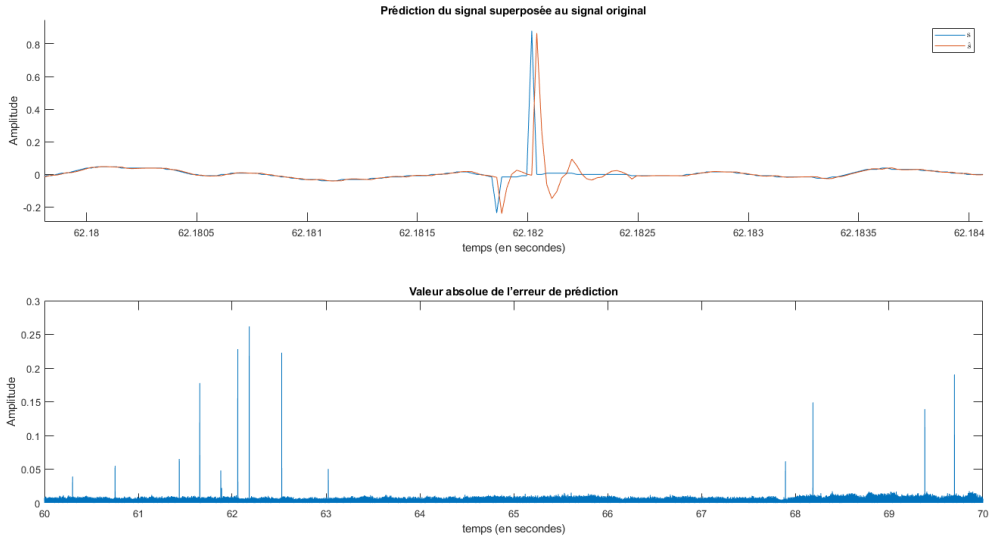

FIGURE 4 – Prédiction linéaire du signal (*rouge*) superposée au signal original (*bleu*) et la valeur absolue de l'erreur de prédiction $\varepsilon_n = \hat{s}_n - s_n$

□

II.5 Restauration par seuillage

Choisissez un seuil Σ pour identifier à partir de l'erreur de prédiction, les indices k_i des dates de craquement. Pour chacun de ces indices, remplacer la valeur s_{k_i} (i.e. craquement) par la valeur médiane (fonction `median.m` sous Matlab) des échantillons de s situés autour de k_i, $\{s_{k_i+l},\ l = -10, -9, \ldots, 0, \ldots 10\}$.

code

```
%% 5 Restauration par seuillage

SSeuillage = s1;
for k = 1:length(SSeuillage)-M
    if abs(epsilon(k))> 0.02
        SSeuillage(k+M-2) = median(s1(k+M-2-10:k+M-2+10));
    end
end

figure()
hold on
plot(t1,s1)
plot(t1,SSeuillage)
title('Signal restaure superposee au signal original')
xlabel('temps (en secondes)')
ylabel('Amplitude')
legend('s','$\hat{s}$','interpreter','latex')
```

□

Afficher le signal ainsi restauré superposé au signal original bruité (faire un zoom sur la même plage de signal qu'à la question II.4).

figures

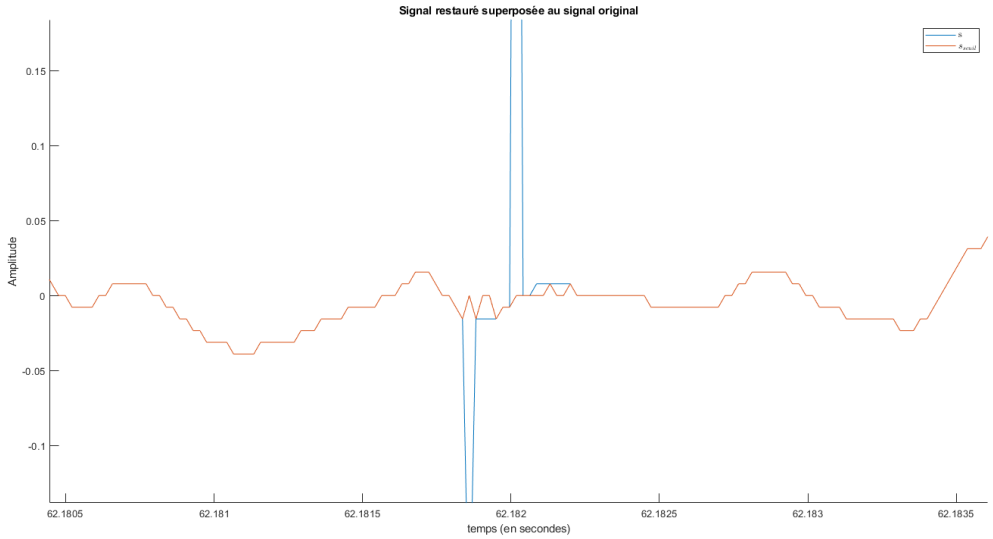

FIGURE 5 – signal restauré par seuillage (*rouge*) superposé au signal original (*bleu*)

□

Écouter le signal restauré. Commentez.

réponse

Le signal n'est pas parfait : on entend encore beaucoup de craquement. Néanmoins, c'est déjà mieux qu'au départ (encore heureux). □

II.6 Restauration par prédiction *causale* / *anticausale*

La prédiction causale (estimation de l'échantillon $s[n]$ à partir des échantillons antérieurs) de la question II.4 produit une erreur de prédiction où chaque craquement est repéré par un pic principal, suivi d'un train de pics secondaires (rebonds dus au temps de réponse du filtre $h[k]$). Le débruitage par seuillage de cette erreur de prédiction peut alors conduire à la détection de faux craquements...Pour éviter cela, on peut procéder à une prédiction causale, combinée à une prédiction anticausale où chaque échantillon $s[n]$ est prédit à partir des M échantillons suivants ($s[n+k]$, $k = 1, \ldots, M$) et ce, en utilisant le même filtre $h[k]$ puisque la fonction d'autocorrélation est paire. Les erreurs de prédiction $\varepsilon_{\text{causale}}$ et $\varepsilon_{\text{anticausale}}$ n'ont alors en commun que les pics principaux localisés aux instants des craquements. Il suffit ensuite de remplacer l'échantillon $s[n_{\text{crack}}]$ par la moyenne des deux prédictions $\widehat{s}_{\text{causal}}[n_{\text{crack}}]$ et $\widehat{s}_{\text{anticausal}}[n_{\text{crack}}]$.

Programmez cette solution en expliquant bien chaque étape de la procédure.

code

```
%% 6 Restauration par prediction causale / anticausale

%Prediction anticausal
SAntiCausale = zeros(length(s1),1); %Creation du vecteur
SAntiCausale(length(s1)-(M-1):length(s1)) = s1(length(s1)-(M-1):length(s1));%M dernieres valeurs egales a celles du signal d origine
 for k = length(s1)-(M+1):-1:1 %M-1 ieme echantillions jusqu au premier
     SAntiCausale(k) = sum(H(1:length(H)).*s1((k+1):(k+M))); %prediction a partir des h[k] trouve precedemment
 end
epsilon1 = abs(SAntiCausale-s1); %erreur de prediction anticausale

%Prediction causal
SCausal = zeros(length(s1),1); %Creation du vecteur
SCausal(1:(M-1)) = s1(1:(M-1));%M premieres valeurs egales e celles du signal d origine
for k = (M+1):1:length(s1) %parcours les M+1 ieme echantillon jusqu au dernier
    SCausal(k) = sum(H(1:length(H)).*s1((k-M):(k-1))); %prediction a partir des h[k] trouve precedemment
end
epsilon2 = abs(SCausal-s1); %erreur de prediction causale

SPrediction = zeros(1,length(t1)); %Vecteur du signal de prediciton causale/anticausale
for k = 1:length(t1(M:end))
    if (epsilon1(k)>0.1) && (epsilon2(k)>0.1) %on compare les erreurs e un seuil
        SPrediction(k) = (SCausal(k)+SAntiCausale(k))/2; %Le signal de prediction est egale a la moyenne des 2 predicitons
    else
        SPrediction(k) = s1(k); %La prediciton est egale au signal d origine
    end
end

figure();
hold on;
plot(t1,s1)
plot(t1,SPrediction)
title('Signal restaure par prediction causale / anticausale')
xlabel('temps (en secondes)')
ylabel('Amplitude')
legend('s','$s_{restaure}$','interpreter','latex')
```

□

Affichez le résultat de la restauration ainsi obtenue.

figures

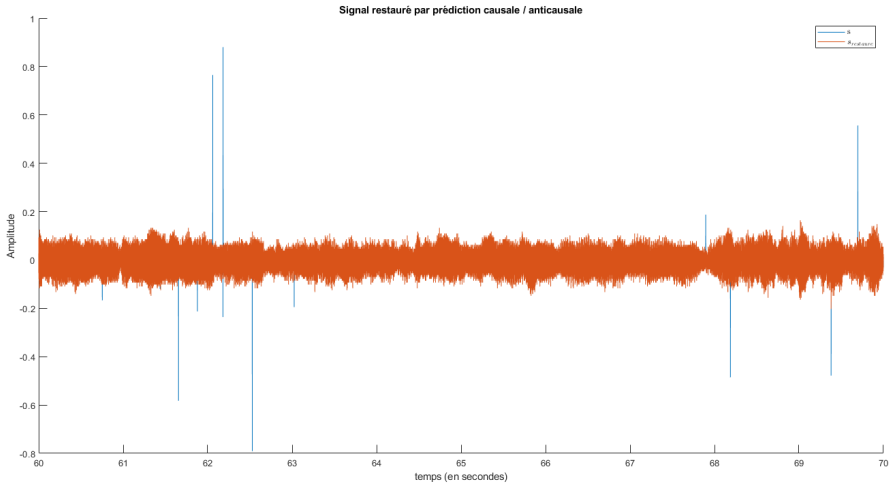

FIGURE 6 – Signal restauré par prédiction causale / anticausale

□

Écouter le signal restauré. Concluez.

réponse

Le signal est bien plus audible : on entend des craquements uniquement lorsque le signal a une très grande amplitude. Comme on peut le voir sur la figure 6, une grande partie des pics ont été supprimés et les seuls restant sont difficilement visible à l'oeil nu. □