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

4 ETI – CPE Lyon

Travaux Pratiques TSA

2020-2021

Noms, Prénoms :

Groupe :

Date :

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

□

I.2

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

réponse

□

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

□

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

□

I.5

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

réponse

□

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

□

figures

□

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

□

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

□

figures

□

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

réponse

□

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

□

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

figures

□

II.4 Prédiction linéaire

En utilisant les coefficients $(h[k],\ k = 1, \ldots, M)$ 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 $(s_{n-k},\ k = 1, \ldots, M)$, pour $n \geq M + 1$.

Reproduisez ci-dessous, le code développé pour effectuer la prédiction et le calcul de l'erreur de prédiction.

code

□

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

□

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

□

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

□

Ecouter le signal restauré. Commentez.

réponse

□

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

□

Affichez le résultat de la restauration ainsi obtenue.

figures

□

Ecoutez le signal restauré. Concluez.

réponse

□