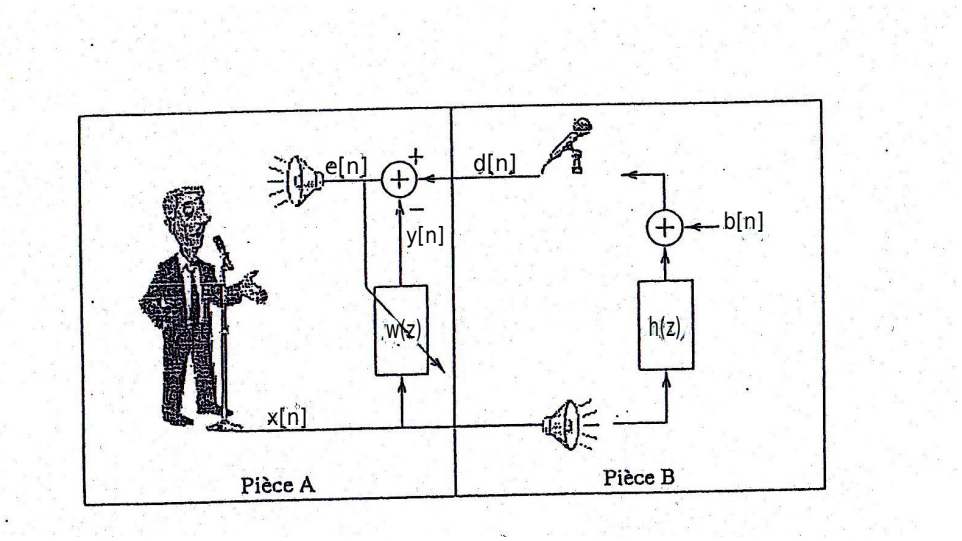
TP2 de Méthodes de Signal Avancées
Annulation d'Écho Acoustique

I. Fijalkow et C. Simon Chane

Compte-rendu attendu à la fin des 4h :
— Codes sources Matlab.
— Résultats de simulations commentées soit à la main sur les figures, soit en pdf.

INTRODUTION



Un exemple simple d'annulation d'écho acoustique est donné dans le cas d'une téléconférence dans laquelle un locuteur parle dans la pièce A et le haut-parleur de la pièce B émet le signal $x\,[n]$. Le microphone de la pièce B reçoit un version filtrée et bruité de $x\,[n]$. Étant directement relié au haut-parleur de la pièce A, le locuteur va donc s'entendre parler. Pour éviter cela, on estime de manière adaptative le filtre h par le filtre w et on envoie sur le haut-parleur de la pièce A uniquement l'erreur commise $e\,[n]$.

I. IMPLÉMENTATION DE L'ALGORITHME LMS

1) **Préparation** Rappeler les équations de l'algorithme LMS.

2) **Génération de signaux test** Dans un premier temps le signal x est un bruit blanc et le signal représentant d est obtenu par filtrage de x par le filtre de réponse impulsionnelle finie $\mathbf{h} = [1\ 0.3\ \ -0.1\ 0.2]^t$ supposé inconnu. Générer x et d.

3) **Mise mise en œuvre de l'algorithme LMS** Écrire une fonction `algolms` qui prend en entrée x, d, P l'ordre du spectre et l'écart-type μ et qui renvoie, pour chaque itération : le filtre w, le signal de sortie y et l'erreur e.

4) **Validation de l'algorithme LMS** Considérer le signal généré au point 2) et utiliser l'algorithme LMS pour calculer le filtre et le signal de sortie.
   — Que doit être $\mathbf{w}_{\text{opt}}$?
   — Tracer la réponse desirée d_n, le signal de sortie y_n et l'erreur e_n dans la même figure en fonction du temps, n.

— Tracer les vrais coefficients et les estimations obtenues par le LMS en fonction du temps, n.

5) **Test de l'algorithme LMS** avec un signal simulé.
Le signal x est un bruit blanc et le signal représentant d est obtenu par filtrage de x plus un bruit. Le filtrage de x est obtenu par le filtre de réponse impulsionnelle finie d'ordre P et fréquence 0.5. Utiliser l'algorithme LMS pour calculer le signal de sortie.
— Étudier l'effet de la longueur P du filtre sur les performances obtenu, $P = 5, 10, 20$.
— Pour P fixé étudier l'effet du choix de μ sur la vitesse de convergence, $\mu = 0.01, 0.1, 0.5$.

II. Application

Mise en œuvre de l'annulation d'écho acoustique :

1) **Signal audio avec une voix**
— Charger le signal audio dans le file `Voix1.wav`.
— Écouter le signal à l'aide de la fonction `sound`.
— Charger Rep.dat, réponse impulsionnelle de la chambre et filtrer le signal.
— Écouter le signal filtré.
— Ajouter un bruit blanc.
— Écouter le signal filtré plus bruit.
— Annuler l'écho en utilisant l'algorithme LMS.
— Comment est le son après l'annulation d'écho ? Jouer sur les paramètres pour voir leur effets.

2) **Signal audio avec deux voix** On considère un interlocuteur dans la pièce A et un interlocuteur dans la pièce B. Le signal au microphone dans la pièce B est donné par le signal "proche" de l'interlocuteur dans la pièce B plus le signal "éloigné" avec écho de l'interlocuteur dans la pièce A. Le but est annuler le signal "éloigné" avec écho pour transmettre seulement le signal "proche".
— Charger le signal "eloigné" : `load farspeech` (il est enregistré dans le vecteur `x`).
— Charger le signal "proche" : `load nearspeech` (il est enregistré dans le vecteur `v`).
— Écouter les signaux à l'aide de la fonction `sound`.
— Charger `Rep.dat`, réponse impulsionnelle de la chambre (c.f. figure 1) et filtrer le signal éloigné. On obtient le signal éloigné avec écho.
— Créer le dialogue entre le signal éloigné filtré et le signal proche, et écouter le dialogue.
— Annuler le signal éloigné avec écho en utilisant l'algorithme LMS.
— Comment est le son après l'annulation d'écho ? Jouer sur les paramètres pour voir leur effets.

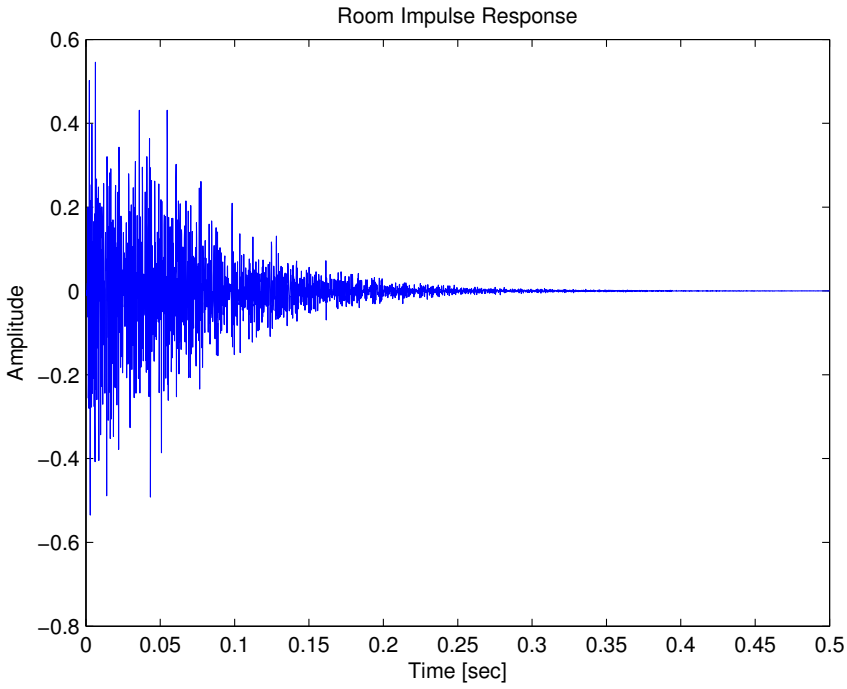


FIGURE 1. Réponse impulsionnelle de la pièce.

AIDE À LA PROGRAMMATION

Commandes Matlab utiles pour réaliser le filtrage :

— Les filtres sont représentés dans le domaine complexe de z par la fonction de transfert suivante :

$$H(z) = \frac{b(1) + b(2)z^{-1} + \cdots + b(n+1)z^{-n}}{a(1) + a(2)z^{-1} + \cdots + a(m+1)z^{-m}},$$

donc pour définir un filtre, il faut connaître les coefficients du numérateur et du dénominateur.
Sous MATLAB, ces coefficients doivent être rangés dans deux vecteurs lignes a et b, puis, pour effectuer le filtrage, il suffit de taper : `y = filter(b,a,X)`. Ceci filtre les données dans le vecteur X par le filtre obtenu par le vecteur b (coefficients du numérateur) et le vecteur a (coefficients du dénominateur).

— La fonction **fir1** `(P-1,Wn)` permet de réaliser un filtre de réponse impulsionnelle finie (FIR) d'ordre p. Elle génère les coefficients b seulement (donc $a = 1$). Il suffit de choisir l'ordre P du filtre et les fréquences rangées dans un vecteur W_n dans l'ordre croissant.

— La fonction `[y, Fs] =` **wavread** `('nomfile.wav')` permet de lire le fichier audio `nomfichier.wav` et elle renvoie le données dans y et la fréquence dans Fs.

— La fonction **sound** `(y, Fs)` nous permet d'écouter le fichier audio y.