TP 1 : Outils d'analyse spectrale des signaux déterministes et aléatoires

Remarque : les questions dont le numéro est suivi du signe † sont des questions théoriques, ne nécessitant pas l'utilisation de Matlab, qui doivent impérativement être résolues avant la séance de TP.

I. Analyse spectrale des signaux déterministes

L'objectif de cette manipulation est d'illustrer les effets de l'utilisation de la Transformée de Fourier Discrète pour l'analyse spectrale des signaux déterministes. Avant d'arriver en séance, vous devez impérativement avoir assimilé les notions théoriques concernant :
– les hypothèses sous-jacentes à l'utilisation de la Transformée de Fourier Discrète,
– les calculs d'erreur en fréquence et d'erreur relative en amplitude,
– les techniques de bourrage de zéros (*zero padding*) et de fenêtrage,
– les problèmes de résolution en fréquence.

I.1. Analyse spectrale d'une sinusoïde

On va effectuer l'analyse spectrale par TFD d'un signal sinusoïdal de différentes fréquences. Comme la TFD nécessite un nombre fini d'échantillons, on va échantillonner ce signal à 1 kHz et en prendre un nombre N = 100 d'échantillons.

1. † Pour quelles fréquences f_0 de la sinusoïde la fréquence d'échantillonnage respecte-t-elle le théorème de Shannon ?
2. † Montrer que, pour un signal de durée finie N, la TFD donne un échantillonnage de la TFSD. En déduire la précision fréquentielle que l'on peut obtenir par TFD.
3. † Dans le cas de la sinusoïde étudiée, à quoi correspond la TFSD et donc la TFD ?
4. Tracer la représentation temporelle d'un tel signal pour f_0 = 100 Hz. Tracer sa TFD. Peut-on ainsi retrouver la fréquence et l'amplitude de la sinusoïde ? L'amplitude maximale de la TFD correspond-elle à l'amplitude de la sinusoïde ? Comment pourrait-on normaliser la TFD pour retrouver cette amplitude ? Peut-on également retrouver sa phase à l'origine ?
5. Faire de même pour f_0 = 95 Hz. Commenter.
6. En déduire l'erreur relative en amplitude obtenue par l'analyse spectrale de la sinusoïde par TFD. Comparer avec la valeur théorique, calculée en TD.

On va maintenant effectuer du bourrage de zéros (*zero padding*), c'est-à-dire prolonger le signal étudié par des zéros avant d'en calculer sa TFD.

7. † Quelle est la relation entre la TFD de ce nouveau signal de longueur $N_0 > N$ et la TFSD du signal original ?
8. Construire un signal 10 fois plus long que le précédent en employant une telle technique et calculer sa TFD. Superposer le module de cette TFD au module de la TFD du signal original. Effectuer cette opération pour f_0 = 100 Hz et f_0 = 95 Hz et commenter.

I.2. Fenêtrage

Afin d'obtenir une meilleure résolution en amplitude, on va effectuer un fenêtrage du signal avant de calculer sa TFD. De façon implicite, nous avions considéré jusqu'alors une fenêtre rectangulaire. On va maintenant étudier le cas de fenêtre de Hanning.

1. † Expliquer l'effet sur la TFSD de la pondération du signal par une fenêtre. En déduire l'effet sur la TFD et donc sur l'analyse spectrale effectuée.
2. † Quelles sont les propriétés souhaitées pour une telle fenêtre ?
3. Tracer la réponse en fréquence des fenêtres rectangulaire et Hanning et les comparer (on pourra par exemple tracer, en échelle logarithmique, le module de leur TFD après bourrage de zéros).
4. Normaliser ces fenêtres afin que dans le meilleur des cas l'amplitude maximale de la TFD soit égale à l'amplitude de la sinusoïde.
5. Fenêtrer le signal sinusoïdal et tracer le module de sa TFD. Commenter suivant la fenêtre prise en compte, pour $f_0 = 100$ Hz et $f_0 = 95$ Hz (avec $N = 100$).
6. Donner la précision relative en amplitude obtenue par Matlab pour l'analyse spectrale de la sinusoïde par TFD suivant la fenêtre prise en compte.

I.3. Résolution spectrale

Soit le signal suivant constitué de deux sinusoïdes :

$$x[n] = A_1 \sin(2\pi n f_1/f_e + \varphi_1) + A_2 \sin(2\pi n f_2/f_e + \varphi_2).$$

L'étude d'un tel signal permet de faire apparaître les problèmes liés à la résolution de l'analyse. On se place dans un cas où la première sinusoïde a une amplitude unité $A_1 = 1$ et une fréquence $f_1 = 95$ Hz, la fréquence d'échantillonnage restant à 1 kHz, le nombre d'échantillons étant également fixé à $N = 100$. On fera varier l'amplitude et la fréquence de la seconde sinusoïde afin de mettre en avant les problèmes rencontrés lors de l'analyse spectrale par TFD de signaux réels.

1. En fixant l'amplitude $A_2 = A_1$, suivant la fenêtre de pondération utilisée, jusqu'à quelle fréquence f_2 peut-elle être abaissée tout en distinguant clairement les deux fréquences ? Quelles sont alors les amplitudes correspondantes ?
2. En fixant la fréquence $f_2 = 140$ Hz, suivant la fenêtre de pondération utilisée, jusqu'à quelle amplitude A_2 peut-elle être abaissée tout en distinguant clairement les deux fréquences ? Quelles sont alors les amplitudes correspondantes ?

II. Analyse spectrale des signaux aléatoires

L'objectif de cette partie est d'étudier l'analyse spectrale des signaux aléatoires. En cours et en TD, on a pu caractériser de façon théorique des estimateurs de la fonction d'autocorrélation et de la densité spectrale de puissance (DSP) de signaux aléatoires. On va maintenant étudier en pratique, sur des signaux classiques simulés, l'utilisation de tels estimateurs.

Avant d'arriver en séance, vous devez impérativement avoir assimilé les notions théoriques concernant :

- les caractéristiques statistiques des estimateurs (biais, variance, …),
- les caractéristiques statistiques d'ordre 2 des signaux aléatoires et en particulier l'autocorrélation et son estimation,
- les caractéristiques fréquentielles des signaux aléatoires stationnaires au sens large et en particulier la densité spectrale de puissance et son estimation.

II.1. Estimation de l'autocorrélation

On rappelle que l'autocorrélation d'un signal réel à temps discret stationnaire $X[n]$ est définie par

$$R_X[m] = E\{X[n+m]X[n]\}$$

D'un point de vue théorique, calculer cette fonction nécessite la connaissance de la loi de probabilité de *X*. En pratique, on ne dispose qu'un nombre fini d'échantillons d'une seule réalisation du signal, notée $x[n]$, n=1…N, et l'on doit estimer l'autocorrélation à partir de ces échantillons.

On connaît deux estimateurs empiriques de l'autocorrélation, le premier est théoriquement non-biaisé :

$$\overset{o}{R}_X^{nb}[m] = \frac{1}{N-m}\sum_{n=1}^{N-m} x[n+m]x[n] \qquad m \geq 0 \qquad \text{et} \qquad \overset{o}{R}_X^{nb}[-m] = \overset{o}{R}_X^{nb}[m]$$

le second est théoriquement biaisé :

$$\overset{o}{R}_X^{b}[m] = \frac{1}{N}\sum_{n=1}^{N-m} x[n+m]x[n] \qquad m \geq 0 \qquad \text{et} \qquad \overset{o}{R}_X^{b}[-m] = \overset{o}{R}_X^{b}[m]$$

On va étudier l'utilisation pratique de ces estimateurs pour l'estimation de l'autocorrélation du signal $X_1[n] = B[n] + 1$, où $B[n]$ est un bruit blanc gaussien centré de variance unitaire.

1. † Rappeler l'autocorrélation théorique de ce signal.
2. † Rappeler les formules de biais pour les estimateurs non-biaisé et biaisé de l'autocorrélation. Quelles sont les valeurs théoriques de ces biais pour le signal ci-dessus ?
3. En utilisant la fonction Matlab *xcorr*, calculer les estimateurs biaisé et non-biaisé de l'autocorrélation pour une centaine de réalisations du signal ci-dessus. Chaque réalisation contient N=50 échantillons temporels. Estimer de façon empirique, pour cette centaine de réalisations, la valeur moyenne de l'estimateur ainsi que sa variance.
4. Tracer sur un même graphique la valeur moyenne ainsi que la valeur moyenne plus ou moins l'écart-type de ces estimateurs.
5. Commenter ces graphiques en termes de biais et variance des différents estimateurs pour les différentes valeurs de *m*.

II.2. Analyse spectrale non-paramétrique

La densité spectrale de puissance (DSP) d'un signal aléatoire est définie comme la transformée de Fourier de sa fonction d'autocorrélation. D'un point de vue empirique, on peut fabriquer deux types d'estimateur de la DSP d'un signal aléatoire : les premiers estimant l'autocorrélation à partir d'une réalisation du signal puis calculant la transformée de Fourier de cette corrélation ; les seconds estimant directement la DSP à partir de la transformée de Fourier de la réalisation du signal. Dans la suite, on s'intéresse surtout à ce deuxième type d'estimateur, à savoir le périodogramme et ses dérivées.

II.2.1. Périodogramme

Le périodogramme est défini par $\overset{o}{S}_X^{p}(f) = \frac{1}{N}\left|\hat{x}(f)\right|^2$ avec $\hat{x}(f) = \sum_{n=1}^{N} x[n]e^{-j2\pi fn}$ la transformée de Fourier de la réalisation du signal.

1. † Rappeler la DSP théorique d'un bruit blanc centré de variance unitaire.
2. † Le périodogramme est-il un estimateur non-biaisé de la DSP ? Quel est le biais de cet estimateur dans le cas général ? Quel est son biais dans le cas particulier d'un bruit blanc ?
3. Ecrire un programme Matlab calculant le périodogramme pour 100 réalisations d'un bruit blanc centré gaussien de puissance unitaire. Chaque réalisation contient N=128 échantillons temporels. Tracer sur un même graphique la valeur moyenne ainsi que la valeur moyenne plus ou moins l'écart-type de cet estimateur. Répéter le test avec 1000 réalisations. Conclure sur le biais du périodogramme d'un bruit blanc.

II.2.2. Périodogramme moyenné

Pour réduire la variance du périodogramme, on peut utiliser l'approche suivante :

- on divise les N échantillons du signal observé en L sections $x_l[n]$, chacune de $M=N/L$ échantillons,
- on calcule le périodogramme sur chaque section,
- on calcule la moyenne des périodogrammes :

$$\overset{o}{S}{}_X^{pm}(f) = \frac{1}{L}\sum_{l=1}^{L} \overset{o}{S}{}_{X_l}^{p}(f) \quad \text{avec} \quad \overset{o}{S}{}_{X_l}^{p}(f) = \frac{1}{M}\left|\sum_{n=0}^{M-1} x_l[n]e^{-j2\pi fn}\right|^2$$

On va étudier l'utilisation pratique du périodogramme moyenné pour l'estimation de la DSP des signaux suivants :

$X_1[n]$: bruit blanc centré de variance unitaire.

$X_2[n] = \cos(2\pi f_0 n+\varphi_0) + \cos(2\pi f_1 n+\varphi_1)$ avec f_0=0.1, f_1=0.15 et φ_0 et φ_1 deux variables aléatoires uniformément distribuées sur $[0, 2\pi]$.

1. Générer une réalisation de N=1024 échantillons de chacun des signaux ci-dessus.
2. Calculer et tracer le périodogramme moyenné de chaque signal avec différentes valeurs de L dans la formule du périodogramme moyenné : L=1 (périodogramme simple), L=16, L=32, L=128. Conclure sur le meilleur choix de L pour estimer la DSP de chacun des deux signaux.

II.2.3. Périodogramme de Welch

Le périodogramme de Welch combine le moyennage et le lissage pour réduire la variance de l'estimateur :

- on divise le signal en L sections de M échantillons,
- chaque section est multipliée par une fenêtre $w[n]$,
- on calcule le périodogramme de chaque section fenêtrée,
- on calcule la moyenne de ces périodogrammes.

On va comparer le périodogramme simple et le périodogramme de Welch pour l'estimation de la DSP du signal ci-dessous :

$X[n] = \cos(2\pi f_0 n+\varphi_0)+B[n]$ avec f_0=0.1, φ_0 une variable aléatoire uniformément distribuée sur $[0, 2\pi]$ et $B[n]$ un bruit blanc de variance 4.

1. Générer une réalisation de N=1024 échantillons de ce signal.
2. Tracer les périodogrammes simple et de Welch de ce signal en utilisant les fonctions *periodogram* et *pwelch* de Matlab (utiliser les versions par défaut[1]). Commenter.

[1] La version par défaut de la fonction *pwelch* choisit L=8 sections avec un chevauchement de 50% et une fenêtre de Hamming.

TP 2 : Identification, débruitage et analyse temps-fréquence

Il est indispensable d'avoir préparé le TP avant la séance.
Les listings des programmes doivent être joints au rapport.
Tous les résultats et courbes doivent être commentés.

1 Présentation du TP

Ce TP est constitué de deux parties :

- Dans la première partie, on cherche à caractériser un filtre inconnu en estimant sa réponse impulsionnelle $h[n]$ ou sa réponse en fréquence $\widehat{h}(f)$ à partir de la sortie $y[n]$ du filtre, éventuellement perturbée, mesurée pour une entrée $x[n]$. On étudiera pour cela des méthodes expérimentales s'appuyant sur l'estimation de la corrélation des signaux. On verra ensuite l'utilisation de ce type de méthodes pour le débruitage des signaux à partir d'une référence de bruit.
- Dans la seconde partie, on illustrera le problème de l'analyse temps fréquence des signaux, en particulier par la transformée de Fourier à fenêtre glissante. Pour cela, on commencera par bien comprendre la signification des paramètres de cette représentation sur des signaux simulés et bien appréhender les problèmes de localisation en temps et en fréquence. Ensuite, on tentera d'analyser des signaux réels.

Les questions dont le numéro est suivi du signe "†" sont des questions théoriques, ne nécessitant pas l'utilisation de Matlab, qui doivent impérativement être résolues avant la séance de TP.

Première partie

2 Identification de la réponse impulsionnelle d'un filtre

On considère un signal $e[n]$ mis en entrée d'un filtre de réponse impulsionnelle $h[n]$. On dispose de l'entrée et d'une sortie perturbée $s[n]$, la perturbation étant notée $b[n]$ (*cf* Figure 1).

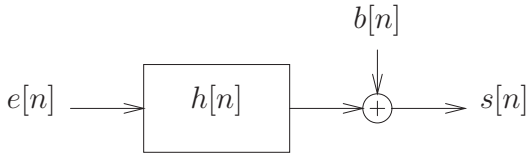


FIGURE 1 – Relation entrée sortie d'un filtre

2.1 Étude du filtre

Dans toute cette partie, on considérera le filtre de fonction de transfert

$$H(z) = \frac{1 - 0,5z^{-1} + z^{-2}}{1 - 0,729z^{-3}}.$$

1. En utilisant la fonction Matlab `freqz`, tracer la réponse en fréquence de ce filtre.
2. Tracer sa réponse impulsionnelle $h[n]$. **Dans la suite du TP on considérera que cette réponse impulsionnelle est finie, et qu'elle peut être estimée par un filtre RIF d'ordre** $P = 50$. Commentez cette approximation...

3† Donner la relation liant l'intercorrélation sortie-entrée $R_{se}[m]$ à l'autocorrélation de l'entrée $R_e[m]$ et à l'intercorrélation perturbation-entrée $R_{be}[m]$. Que devient cette relation si $e[n]$ et $b[n]$ sont non-corrélés et que $b[n]$ est centré ?

4† La relation de la question 3 peut s'écrire sous forme matricielle $\Gamma h = c$, avec Γ matrice carrée, en rangeant les coefficients de la réponse impulsionnelle dans un vecteur colonne. A quoi correspondent les éléments de la matrice Γ et du vecteur c ?

5. Dans une expérience, on a placé un signal $e[n]$ à l'entrée du système de Fig. 1 et on a mesuré la sortie $s[n]$. Les signaux $e[n]$ et $s[n]$ sont disponibles dans le fichier *sig.mat*. Construire la matrice Γ et le vecteur c en utilisant ces signaux et les corrélations estimées. Résoudre le système pour estimer la réponse impulsionnelle du filtre et sa réponse en fréquence. Tracer ces réponses et les comparer aux réponses théoriques. Conclusions...

6† Comment s'écrit la relation de la question 3 dans le domaine fréquentiel (on notera S_{se}, S_e et S_{be} les densités spectrales et inter-spectrales utilisées) ?

7† Comment peut-on estimer la réponse en fréquence du filtre grâce à cette dernière relation ? Quelles sont les conditions nécessaires pour que cette estimation se passe *sans problèmes* ?

8. Utiliser cette méthode pour estimer la réponse en fréquence puis la réponse impulsionnelle du filtre. Tracer ces réponses et les comparer aux réponses théoriques. Conclusions...

3 Débruitage d'un signal avec référence de bruit

On va exploiter une des méthodes précédentes pour identifier la réponse impulsionnelle d'un filtre dans une configuration de débruitage d'un signal avec référence de bruit telle

que représentée Figure 2. On suppose disposer de deux capteurs : le premier mesure le signal bruité (Mesure) et le deuxième mesure la référence de bruit b[n].

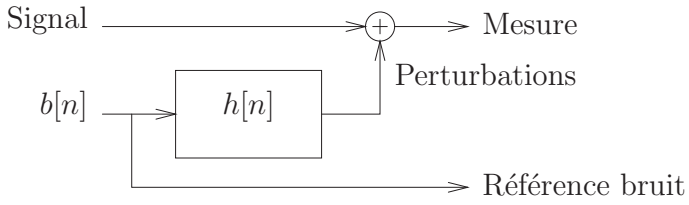


FIGURE 2 – Configuration pour le débruitage avec référence de bruit

1† Montrer que l'on peut bien exploiter la corrélation pour débruiter le signal.

Dans la suite, on applique la méthode ci-dessus dans deux applications différentes.

3.1 Application biomédicale

Le premier exemple est une application biomédicale qui concerne la mesure de l'électrocardiogramme d'un fœtus dans le ventre de sa mère. Un capteur, situé sur le ventre de la mère donne le signal électrique du cœur du fœtus (signal) additionné au signal électrique issu du cœur de la mère (perturbations). Un autre capteur situé près du cœur de la mère donne le signal électrique du cœur de la mère (référence de bruit). On modélise la relation entre la référence de bruit et les perturbations par un système linéaire de réponse impulsionnelle $h[n]$. Une fois estimée la réponse impulsionnelle $h[n]$ il est aisé de débruiter le signal à partir de la référence de bruit.

Choisissez parmi les méthodes précédentes exploitant la corrélation celle qui vous parait donner les meilleurs résultats. Les mesures `mes`, la référence de bruit `ref` et le signal `sig` sont disponibles dans le fichier `estim.mat`.

1. Estimer et tracer la réponse impulsionnelle du filtre.
2. A partir de la réponse impulsionnelle estimée, en déduire une estimation du signal débruité. Tracer et comparer les signaux estimé et original. Conclusions...

3.2 Application audio

Le deuxième exemple est une application audio. On suppose que le signal utile (signal) est la voix d'une personne qu'on veut enregistrer avec un microphone "A" situé près de cette personne. En pratique, cet enregistrement est pollué par une perturbation, résultat de la propagation d'une source de bruit lointain et fort jusqu'au microphone "A". En modélisant la propagation de ce bruit par un effet de filtrage, nous pouvons utiliser la technique ci-dessus en plaçant un deuxième microphone "B" près de la source du bruit (référence de bruit). On retrouve donc le modèle de la figure ci-dessus [1].

1. On suppose ici que le signal utile est faible devant le bruit. Si ce n'est pas le cas, on doit également tenir compte de la propagation du signal utile jusqu'au microphone B, ce qui nécessite l'utilisation des techniques de débruitage plus sophistiquées.

Le fichier z2.wav contient un signal sonore enregistré par le microphone “A”. Le fichier b2.wav contient une mesure de ce bruit par le microphone “B”.

1. Ecouter ces deux signaux.
2. En utilisant la méthode expliquée ci-dessus, estimer la réponse impulsionnelle du filtre de propagation et en déduire le signal utile.
3. Ecouter le signal utile.

Deuxième partie

4 Analyse temps-fréquence

L'analyse temps-fréquence a été abordée en cours au travers de la transformée de Fourier à fenêtre glissante (ou TF à court terme – *short term Fourier Transform* en anglais). L'idée de base de cette représentations est de décomposer le signal sur une famille de sinusoïdes fenêtrées.
Pour une fenêtre $g(t)$ donnée, les coefficients d'une telle décomposition se calculent par :

$$C_x(t,f) = \int_{-\infty}^{+\infty} x(\tau) g^*_{t,f}(\tau) \mathrm{d}\tau \text{ avec } g_{t,f}(\tau) = g(\tau - t) e^{2j\pi f \tau}$$

Il est alors immédiat de montrer que :

$$C_x(t,f) = \int_{-\infty}^{+\infty} x(\tau) g^*(\tau - t) e^{-2j\pi f \tau} \mathrm{d}\tau = \mathrm{TF}\{x(\tau) g^*(\tau - t)\}$$

d'où le nom de Transformée de Fourier à fenêtre glissante ($g^*(\tau - t)$ correspondant à une fenêtre centrée en t). En pratique, il est procédé à une discrétisation de cette relation en temps et en fréquence et le calcul est effectué par Transformée de Fourier Rapide (FFT). Le spectrogramme d'un signal est défini comme le module au carré de la TF à court terme. Dans une telle représentation, il se pose la question de localisation en temps et en fréquence :

1† Pour une fenêtre de support temporel ΔT et de support fréquentiel ΔF, donnez le domaine du plan temps-fréquence influencé par un événement intervenant à un instant t_0.

2† Pour une telle fenêtre, donnez le domaine du plan temps-fréquence influencé par un événement de fréquence f_0.

3† Sur quels paramètres de la fenêtre faut-il jouer pour obtenir une meilleure localisation en temps ? en fréquence ? Peut-on avoir simultanément une bonne localisation en temps et en fréquence ?

4.1 Compréhension sur des signaux simulés

Il va s'agir principalement ici de bien comprendre l'utilisation des programmes calculant le spectrogramme (voir l'annexe sur l'utilisation de la fonction `spectrogram` de Matlab). Pour tous les signaux générés, on prendra une fréquence d'échantillonnage de $F_e = 8$ kHz et un nombre d'échantillons de $N = 1024$.

1. Tracer le spectrogramme d'une impulsion unité (Kronecker) pour différents types de fenêtres. Conclure sur la localisation temporelle de la TF à court terme suivant la fenêtre...
2. Tracer le spectrogramme d'une sinusoïde (faire varier la fréquence) pour différents types de fenêtres. Conclure sur la localisation fréquentielle de la TF à court terme suivant la fenêtre et la fréquence...
3. Tracer le spectrogramme de signaux comportant à la fois des événements ponctuels et des fréquences pures. Conclure sur le compromis à régler entre localisation temporelle et fréquentielle...
4. Poursuivre par l'analyse de signaux synthétiques de votre choix (chirp, sinusoïdes par morceaux, signaux bruités, etc.) Commentez vos résultats...

4.2 Analyse de signaux réels

Vous disposez d'un ensemble de fichiers de signaux qu'il vous faut analyser. Pour cela, vous pouvez tracer leur représentation temporelle, leur représentation fréquentielle, mais également leur spectrogrammes. Commentez vos résultats d'analyse sur quelques-uns de ces signaux...

5 Annexe

De nombreuses fonctions Matlab vous seront utiles dans ce TP.

Dans la première partie, les fonctions `randn`, `freqz`, `filter`, `xcorr` et `toeplitz` pourront être utilisées. Il est prudent de faire appel à l'aide en ligne avant de les utiliser...

Dans la deuxième partie, l'analyse temps-fréquence sera effectuée grâce à la fonction `spectrogram` de Matlab décrite ci-dessous.

5.1 Annexe : Fonction Matlab `spectrogram`

```
spectrogram Spectrogram using a Short-Time Fourier Transform (STFT).
   S = spectrogram(X) returns the short-time Fourier transform of the
   signal specified by vector X in the matrix S. By default, X is divided
   into eight segments with 50\% overlap, and each segment is windowed with
   a Hamming window. The number of frequency points used to calculate the
   discrete Fourier transforms is equal to the larger of 256 or the next
   power of two greater than the segment length.
```

```
    S = spectrogram(X,WINDOW), when WINDOW is a vector, divides X into
    segments of the same length as WINDOW, and then windows each segment
    with the vector specified in WINDOW.  If WINDOW is an integer, the
    function divides X into segments of length equal to that integer value
    and windows each segment with a Hamming window.  If WINDOW is not
    specified, the default is used.

    S = spectrogram(X,WINDOW,NOVERLAP) specifies NOVERLAP samples of
    overlap between adjoining segments. NOVERLAP must be an integer smaller
    than WINDOW if WINDOW is an integer.  NOVERLAP must be an integer
    smaller than the length of WINDOW if WINDOW is a vector.  If NOVERLAP
    is not specified, the default value is used to obtain a 50\% overlap.

    S = spectrogram(X,WINDOW,NOVERLAP,NFFT) specifies the number of
    frequency points used to calculate the discrete Fourier transforms.
    If NFFT is not specified, the default NFFT is used.

    S = spectrogram(X,WINDOW,NOVERLAP,NFFT,Fs) specifies the sample rate,
    Fs, in Hz. If Fs is specified as empty, it defaults to 1 Hz. If it is
    not specified, normalized frequency is used.

spectrogram(...) with no output arguments plots the PSD estimate for
    each segment on a surface in the current figure.
```

TP3 : Analyse, codage et synthèse de la parole

** Ce TP est fortement basé sur les cours et les TD associés. Il est donc indispensable de réviser les cours et les TD avant la séance.*
** Les questions marquées du signe † doivent être résolues avant la séance.*

I. Analyse

Le fichier *sig.wav* contient un signal de parole correspondant à l'enregistrement du mot « assécher », échantillonné à Fe=11 kHz. Charger ce signal dans Matlab en utilisant la fonction *audioread* et l'écouter.

1. Tracer la représentation temporelle (audiogramme) de ce signal. Pouvez-vous distinguer les 5 phonèmes ? Créer un nouveau signal *x* en supprimant les zones de silence avant et après le mot dans le signal d'origine. En zoomant sur chaque phonème, vérifier s'il s'agit d'un son voisé on non-voisé. Délimiter chaque phonème.

2. Tracer le module de la transformée de Fourier du signal *x* et commenter le résultat.

3. Tracer le spectrogramme à bande étroite du signal (calculé sur les tranches de 0.05 sec) en utilisant la fonction Matlab ci-dessous :

```
spectrogram(x,0.05*Fe,0.025*Fe,0.05*Fe,Fe,'yaxis');
```

Peut-on estimer approximativement la période pitch du signal à partir de ce spectrogramme ?

4. Tracer le spectrogramme à bande large du signal (calculé sur les tranches de 0.01 sec) en utilisant la fonction Matlab ci-dessous :

```
spectrogram(x,0.01*Fe,0.005*Fe,0.01*Fe,Fe,'yaxis');
```

Peut-on distinguer les 5 phonèmes sur ce spectrogramme ? Peut-on en conclure la similitude entre le troisième et le cinquième phonèmes ?

II. Codage LPC

Dans la suite, on divise le signal *x* en tranches de 30 millisecondes (non-recouvrantes) et on estime les paramètres du modèle LPC de la production du signal de parole pour ces tranches.

Pour décider si une tranche du signal est voisée ou non-voisée, on peut en pratique se baser sur des critères tels que la puissance moyenne ou le taux de passages par zéro. Pour gagner du temps, on n'utilisera pas ce genre de critère dans la suite. On se basera plutôt sur nos informations a priori : si la tranche du signal appartient aux 2ème ou 4ème phonèmes, elle est non-voisée, sinon elle est voisée.

5. Estimer la puissance moyenne de chaque tranche et stocker le résultat dans un vecteur.

6. Si la tranche est voisée, estimer sa période pitch en estimant la fonction d'autocorrélation de la tranche avec la version biaisée de la fonction Matlab *xcorr* et en identifiant la position du pic de cette fonction dans l'intervalle des valeurs possibles de la période pitch (de 1/600 à 1/70 sec). Si la tranche est non-voisée, on associera une valeur nulle au pitch. Stocker le résultat dans un vecteur.

Le modèle autorégressif de signal de parole est défini par l'équation ci-dessous :

$$x(n)+a_1x(n-1)+a_2x(n-2)+\ldots+a_Px(n-P)=e(n)$$

7.† Quelle est la fonction de transfert de ce modèle ?

8.† En suivant l'approche présentée en TD, montrer que les coefficients a_i dans le modèle ci-dessus (appelés *coefficients LPC*) peuvent être estimés en résolvant l'équation matricielle de la page suivante où les échantillons de la fonction d'autocorrélation pourraient être calculés par la fonction Matlab *xcorr*. La matrice étant une matrice de Toeplitz (à diagonales constantes), elle peut être construite à partir de sa première ligne à l'aide de la fonction Matlab *toeplitz*.

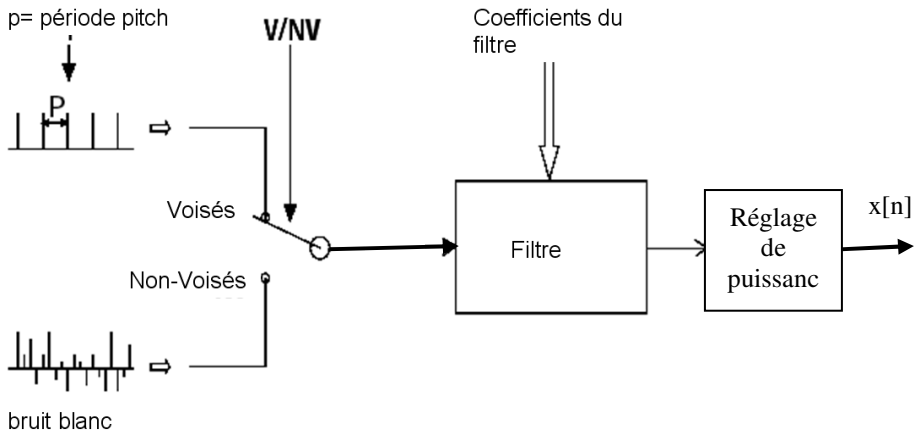
$$\begin{pmatrix} R_x(0) & R_x(1) & \cdots & R_x(P-1) \\ R_x(1) & R_x(0) & \cdots & R_x(P-2) \\ \vdots & \vdots & \cdots & \vdots \\ R_x(P-1) & R_x(P-2) & \cdots & R_x(0) \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_P \end{pmatrix} = \begin{pmatrix} -R_x(1) \\ -R_x(2) \\ \vdots \\ -R_x(P) \end{pmatrix}$$

9. Calculer les coefficients LPC de chaque tranche pour un modèle d'ordre P=12 et stocker le résultat dans une matrice.

10. Si la puissance moyenne, la période pitch et chacun des coefficients LPC sont codés sur 8 bits, il faut combien de bits pour transmettre toutes les informations du signal entier ? Comparer avec le nombre de bits nécessaires pour envoyer le signal sans codage si chaque échantillon est codé sur 8 bits. Conclusion ?

III. Synthèse

On utilise le modèle ci-dessous pour synthétiser chaque tranche du signal de parole à partir des informations extraites dans la section précédente. Dans ce modèle, la source est un train d'impulsions de période pitch si la tranche est voisée et un bruit blanc (généré par la fonction Matlab *randn*) si elle est non-voisée. Le filtrage du signal source par le modèle autorégressif peut être effectué en utilisant la fonction Matlab *filter*.



11. Synthétiser le signal et écouter le résultat. Conclusion ?

12. Recommencer la synthèse en multipliant la période pitch de toutes les tranches par 2. Que constatez-vous ?

13. Recommencer en fixant une période pitch constante pour toutes les tranches voisées. Conclure sur l'importance de l'information pitch.

14. A partir de votre réponse à la question 7, calculer la réponse en fréquence du modèle AR. Si on suppose que $e(n)$ est un bruit blanc (le cas des phonèmes non-voisés), quelle est l'expression de la DSP d'une tranche du signal $x(n)$? Utiliser les coefficients du modèle AR identifiés pour tracer la racine carrée de cette estimation de DSP (à un facteur d'échelle près) pour 2 tranches du signal appartenant aux 2 phonèmes non-voisés. Pour ce faire, vous pouvez utiliser la fonction freqz de Matlab.

IV. Améliorations

Dans les parties analyse et synthèse, on a utilisé des fenêtres rectangulaires non-recouvrantes pour diviser le signal en plusieurs tranches de courte durée, puis les reconcaténer à la fin. Pour améliorer la qualité du signal synthétisé, on utilise souvent des fenêtres de Hanning partiellement recouvrantes pour l'analyse et la synthèse.

15. Modifier votre programme en utilisant des fenêtres de Hanning de 30 millisecondes avec un recouvrement de 50% entre deux fenêtres successives. Ecouter le signal synthétisé. Est-il de meilleure qualité ? Pourquoi ? Quel est le prix à payer ?

Annexe : Fonctions Matlab utilisées

spectrogram Spectrogram using a Short-Time Fourier Transform (STFT).

S = spectrogram(X) returns the spectrogram of the signal specified b vector X in the matrix S. By default, X is divided into eight segments with 50% overlap, each segment is windowed with a Hamming window. The number of frequency points used to calculate the discrete Fourier transforms is equal to the maximum of 256 or the next power of two greater than the length of each segment of X.

S = spectrogram(X,WINDOW) when WINDOW is a vector, divides X into segments of length equal to the length of WINDOW, and then windows each segment with the vector specified in WINDOW. If WINDOW is an integer, X is divided into segments of length equal to that integer value, and a Hamming window of equal length is used. If WINDOW is not specified, the default is used.

S = spectrogram(X,WINDOW,NOVERLAP) NOVERLAP is the number of samples each segment of X overlaps. NOVERLAP must be an integer smaller than WINDOW if WINDOW is an integer. NOVERLAP must be an integer smaller than the length of WINDOW if WINDOW is a vector. If NOVERLAP is not specified, the default value is used to obtain a 50% overlap.

S = spectrogram(X,WINDOW,NOVERLAP,NFFT) specifies the number of frequency points used to calculate the discrete Fourier transforms. If NFFT is not specified, the default NFFT is used.

S = spectrogram(X,WINDOW,NOVERLAP,NFFT,Fs) Fs is the sampling frequency specified in Hz. If Fs is specified as empty, it defaults to 1 Hz. If it is not specified, normalized frequency is used.

spectrogram(...) with no output arguments plots the PSD estimate for each segment on a surface in the current figure.

filter One-dimensional digital filter.

Y = filter(B,A,X) filters the data in vector X with the filter described by vectors A and B to create the filtered data Y. The filter is a "Direct Form II Transposed" implementation of the standard difference equation:

$$a(1)*y(n) = b(1)*x(n) + b(2)*x(n-1) + ... + b(nb+1)*x(n-nb) - a(2)*y(n-1) - ... - a(na+1)*y(n-na)$$

freqz Frequency response of digital filter.

[H,W] = freqz(B,A,N) returns the N-point complex frequency response vector H and the N-point frequency vector W in radians/sample of the filter:

$$H(e^{jw}) = \frac{B(e^{jw})}{A(e^{jw})} = \frac{b(1) + b(2)e^{-jw} + ... + b(m+1)e^{-jmw}}{a(1) + a(2)e^{-jw} + ... + a(n+1)e^{-jnw}}$$

given numerator and denominator coefficients in vectors B and A.

[H,F] = freqz(...,N,Fs) and [H,F] = freqz(...,N,'whole',Fs) return frequency vector F (in Hz), where Fs is the sampling frequency (in Hz).

freqz(...) with no output arguments plots the magnitude and unwrapped phase of the filter in the current figure window.