TP 1 : Numérisation des signaux

Il est indispensable d'avoir préparé le TP avant la séance, c'est-à-dire :

- **avoir lu le sujet de TP et avoir revu le cours et les travaux dirigés correspondants ;**
- **avoir répondu aux questions théoriques, dont le numéro est suivi du signe † et avoir déposé votre préparation sur l'espace Moodle (une préparation par binôme) ;**
- **avoir étudié les documents « Matlab en bref »et « Matlab pour le traitement du signal »et si possible avoir effectué la !séance d'auto-formation à Matlab *MatlabOnRamp*[1]. Notez que le sujet et son annexe vous aideront sur certains aspects de la programmation en Matlab.**

Les listings de tous les programmes doivent être déposés sur l'espace Moodle en fin de séance.

I Introduction au TP

Vous allez effectuer ce TP par binôme en salle I3 du bâtiment 3TP2. Les enseignants vous fourniront un numéro de compte (du type `1asri_xy`) et le mot de passe. Vous pourrez alors accéder au serveur de fichier de la salle en cliquant sur l'application `ELECTRE` du bureau qui fera apparaître votre espace disque (du type `1asri_xy`). Cet espace vous est réservé, vous pouvez y télécharger des fichiers depuis Moodle et créer vos propres fichier... Cet espace disque est accessible depuis tout ordinateur de la salle...

Vous devez travailler sur cet espace disque et en aucun cas sur le PC.

L'objectif de cette manipulation est d'illustrer les effets de la numérisation des signaux (échantillonnage et quantification).
Avant d'arriver en séance, vous devez impérativement avoir assimilé les notions théoriques concernant :

- l'échantillonnage et en particulier le théorème de Shannon ;
- la quantification ;

II Échantillonnage

Un grand nombre de signaux varient continument au cours du temps et leur stockage sur un support numérique nécessite de ne prendre en compte que des valeurs prises par le signal à certains instants : il s'agit de l'échantillonnage. On ne s'intéressera ici qu'au cas de l'échantillonnage régulier, c'est-à-dire que les instants où l'on mesure la valeur du signal $x(t)$ sont régulièrement espacés dans le temps : $x[n] = x(nT_e)$, où T_e est la période d'échantillonnage.

1. proposée en ligne sur
`https://matlabacademy.mathworks.com/fr/details/matlab-onramp/gettingstarted`

II.1 Un peu de théorie

1† Rappeler l'effet en fréquence de l'échantillonnage temporel d'un signal et les conditions nécessaire pour une reconstruction du signal original.

2† Que se passe-t-il si ces conditions ne sont pas satisfaites ? Que doit on faire en pratique pour éviter des désagréments ? Rappeler le principe d'un filtre anti-repliement...

3† Quelle est la plus haute fréquence contenue dans un signal échantillonnée à la fréquence F_e ?

4† A partir d'un signal $x[n]$ échantillonné à la fréquence F_e, on construit le signal $y[n] = x[S \cdot n]$ avec S entier. Cette opération s'appelle le sous-échantillonnage d'un facteur S. Quelle est la fréquence d'échantillonnage du signal sous-échantillonné $y[n]$? Quel est l'effet en fréquence d'un tel sous-échantillonnage ?

II.2 Illustration de l'échantillonnage sur des signaux audio

On va illustrer les effets de l'échantillonnage sur des signaux sonores. De tels signaux sont disponibles dans des fichiers au format *wave* (`*.wav`) et peuvent être chargés dans Matlab à l'aide de la fonction Matlab `audioread` (cf. annexe). La commande `PlaySignal` (cf. annexe) permet d'écouter un signal sonore contenu dans un vecteur, en effectuant éventuellement un sous-échantillonnage et un filtrage anti-repliement.

5. Charger le signal contenu dans le fichier `signal0.wav`. Quelle est la fréquence d'échantillonnage de ce signal et sur combien de bits a-t-il été codé ? Écouter le signal...
6. Écouter le signal sous-échantillonné d'un facteur S pour différentes valeurs de S comprises entre 2 et 10. Décrire brièvement de façon qualitative le signal restitué suivant les valeurs prises par S. À partir de quelle valeur de S entend-on que le signal a subi des modifications ? Est-il évident que ces modifications sont produites par le repliement spectral ?
7. Écouter le signal filtré avec un filtre anti-repliement avant de le sous-échantillonner d'un facteur S pour différentes valeurs de S. Comparer de façon qualitative la qualité de restitution suivant les valeurs prises par S avec et sans filtre anti-repliement.
8. Tracer la représentation fréquentielle du signal. Vous pourrez par exemple pour cela représenter le module de la Transformée de Fourier Discrète des échantillons du signal en fonction de la fréquence sur l'intervalle $[0, F_e]$:
   `L = length(x); freq = (0:L-1)/L*Fe; X = fft(x); plot(freq,abs(X));`
   Si vous préférez afficher la représentation fréquentielle sur l'intervalle $[-F_e/2, F_e/2]$, il suffit de modifier l'affichage :
   `freq = ((0:L-1)-floor(L/2))/L*Fe; plot(freq,fftshift(abs(X)));`
   D'après le théorème de Shannon, à quelle fréquence ce signal peut-il être échantillonné sans perte d'information ? Cela vous paraît-il cohérent avec vos réponses à la question précédente ?
9. Répondre aux même questions pour le signal contenu dans le fichier `signal1.wav`.

III Quantification

En pratique, un signal peut avoir des valeurs continues et son stockage sur un support numérique ne permet de prendre en compte qu'un nombre fini de valeurs discrètes, d'où la quantification. On va étudier ici l'effet de cette quantification. Notons que l'on s'intéressera ici uniquement aux problèmes liés à la quantification et non aux problèmes de dépassement. La fonction Matlab `quanti` (cf. annexe) permet d'effectuer la quantification d'un signal `x` avec N bits (c'est-à-dire sur 2^N valeurs).

1. Pour différentes valeurs de N, écouter le résultat de la quantification du signal contenu dans `SignalQuantif.mat` (variables `x` – signal – et `Fe` – fréquence d'échantillonnage – que l'on peut récupérer avec la commande `load SignalQuantif.mat`). Décrire de façon qualitative la qualité de restitution.

D'un point de vue théorique on assimile l'erreur de quantification comme bruit, modélisé statistiquement par une suite de variable aléatoires indépendantes uniformément distribuées sur l'intervalle $]-q/2\ ;\ q/2]$ (où q est le *pas de quantification*).

2. Rappeler la relation entre le rapport signal sur bruit de quantification et le nombre de bits établi en utilisant le modèle précédent.
3. Après avoir écouté le signal quantifié pour différentes valeurs de N, la modélisation de l'erreur de quantification vous paraît-elle cohérente ou est elle limitée à une certaines plage de valeurs de N ?

IV Échantillonnage et reconstruction d'un signal simple

Après avoir entendu les effets de la numérisation sur un signal sonore, on va travailler sur un signal d'école afin d'illustrer l'échantillonnage, la reconstruction, et l'application pratique du théorème de Shannon. Le signal `x` est présent dans le fichier `SignalReconst.mat` (avec pour temps associé `t`) et peut être récupéré avec `load SignalReconst.mat`. Ce signal a été échantillonné à une fréquence très élevée (on travaillera par la suite en fréquence normalisée, c'est-à-dire que l'on considère $F_e = 1$) et jouera le rôle de *signal analogique*.

1. Tracer les représentations temporelle et fréquentielle du signal `x`.
2. Simuler l'échantillonnage d'un signal continu en ne conservant qu'un échantillon sur S de `x` (sous-échantillonnage d'un facteur S) **et en mettant les autres échantillons à zéro**. Cela se fait aisément avec Matlab en initialisant un vecteur à zéro et en remplissant un échantillon sur S :
   `y = zeros(size(x)); y(1:S:end) = x(1:S:end);`
   Tracer les représentations temporelle et fréquentielle des signaux résultants. Commenter les résultats suivant les valeurs de S...
3. Quelle est la fréquence minimale avec laquelle on peut échantillonner le signal ? A quel facteur de sous-échantillonnage correspond-elle ? Qu'observe-t-on si l'on prend un facteur de sous-échantillonnage trop grand ? Que doit-on faire en pratique, lorsque la fréquence d'échantillonnage est fixée, pour éviter un tel phénomène ?

Le théorème de Shannon nous donne des conditions sur la fréquence d'échantillonnage afin de ne pas perdre d'information lors de l'échantillonnage et assurant une reconstruction exacte du signal continu. On va donc maintenant essayer de reconstruire le signal que l'on a sous-échantillonné.

4. Pour un sous-échantillonnage d'un facteur S, quelle est la réponse en fréquence du filtre idéal permettant de reconstruire le signal continu ?
5. Construire une telle réponse en fréquence échantillonnée au même pas que la TFD. Pour cela on utilisera des tests vectoriels en Matlab du type `v>a` permet d'obtenir un vecteur de même taille que `v` contenant un 1 ou un 0 selon que l'élément correspondant de `v` est supérieur à `a`. Effectuer le filtrage dans le domaine fréquentiel et revenir dans le domaine temporel par TFD inverse. Comparer le signal reconstruit au signal original. Commenter pour différentes valeurs de S...
6. Conclure sur l'application pratique du théorème de Shannon...

V Annexe

V.1 Fonction Matlab `audioread`

Cette fonction lit un signal dans un fichier au format *wave* et le stocke dans un vecteur. Voir l'aide en ligne pour une aide plus complète...

```
audioread Read audio files
   [Y, FS]=audioread(FILENAME) reads an audio file specified by the
   character vector or string scalar FILENAME, returning the sampled data
   in Y and the sample rate FS, in Hertz.

   See also audioinfo, audiowrite
```

V.2 Fonction `PlaySignal`

Cette fonction permet d'écouter un signal stocké dans un vecteur en effectuant éventuellement un sous échantillonnage d'un facteur S entier, avec ou sans application au préalable d'un filtre anti-repliement.

```
PlaySignal Play sound
   PlaySignal(x,Fs) sends the signal in vector x with sample frequency
   of Fs Hertz to audio device.

   PlaySignal(x,Fs,S) sends the S time sub-sampled signal in vector x

   PlaySignal(x,Fs,S,1) sends the S time sub-sampled signal in vector x
   filtered with an Anti-Aliasing filter to audio device.
```

V.3 Fonction Matlab `quanti`

Cette fonction effectue la quantification d'un signal sur N bits.

```
QUANTI Quantisation of a signal
  y = quanti(x,N) return the signal x with values approximated coding it
  with N bits (quantization).
```