SIG- TP2 :

Compression et Débruitage sonore

Partie 1 : Représentation Temps-Fréquence

Un spectrogramme est une représentation de la puissance spectrale associée aux fréquences au cours du temps. Il est obtenu en faisant des Transformées de Fourier successives sur des fenêtres de courte durée.

1/ Vous disposez d'un signal sonore de bonne qualité. Chargez-le dans Matlab et récupérez sa fréquence d'échantillonnage avec la fonction 'audioread'. Affichez-le et déterminez la fréquence maximale grâce au critère de Shannon.

2/ Représentez le signal sous forme d'un spectrogramme. On prendra 256 comme taille de fenêtre, avec un recouvrement de 50 % et 256 pour le nombre de points de la FFT. Quel est le lien entre la fréquence d'échantillonnage du signal et la fréquence maximale affichée sur le spectrogramme ?

Sous-échantillonnage

3/ Générez un signal sous-échantillonné par un facteur 2 et par un facteur 4. Indice : il existe certainement une fonction déjà disponible sous Matlab. Enregistrez-les avec la fonction 'audiowrite'. Que remarquez-vous lors de l'écoute ?

4/ Représentez sous forme de spectrogrammes les extraits sous-échantillonnés.

5/ Quel traitement simple pouvez-vous mettre en œuvre pour contrer ce phénomène ? Représentez sous forme de spectrogramme les signaux après traitement.

6/ Calculez le taux de compression.

Partie 2 : Débruitage sonore par soustraction spectrale

On s'intéresse ici à la question du débruitage de parole lorsqu'une seule voie d'observation est disponible. Nous commencerons par ajouter de manière artificielle un bruit blanc (présent sur toutes les fréquences) gaussien.

Equation 1 :

$$signal_{bruité}(t) = signal(t) + bruit(t)$$

Equation 2 :

$$|S(f)| = \begin{cases} TF(signal_{bruité}(t))^{\delta} - \alpha DSPM(f)^{\delta} & si\ |S(f)| > \beta . DSPM(f) \\ \beta . DSPM(f) & si\ |S(f)| < \beta . DSPM(f) \end{cases}$$

où $DSPM$ représente la densité spectrale de puissance moyenne du bruit.

Equation 3 :

$$\hat{S}(f) = |S(f)| . e^{i\varphi_x(f)}$$

Equation 4 :

$$signal_{débruité}(t) = TF^{-1}[\hat{S}(f)]$$

Le traitement sera effectué en introduisant deux fenêtres de Hamming avec un recouvrement de 50%. Ces fenêtres prendront en compte des blocs de 256 points chacune.

Une fenêtre de Hamming est une courbe en « cloche » centrée sur le milieu du bloc et vérifiant l'équation 5.

Equation 5 :

$$h(t) = \begin{cases} 0.54 - 0.46\cos\left(\frac{2\pi t}{T}\right) si\ t \in [0, T] \\ 0 \end{cases}$$

Bruitage du signal

1/ Créez un bruit blanc de la taille du signal original.

2/ Représentez sur la même figure le signal original, le bruit et le signal bruité. Ecrivez ce signal bruité dans un nouveau fichier wav.

3/ Utilisez la fonction 'DensSpecPuiss' pour calculer la densité spectrale du bruit. Affichez-la sur une échelle de fréquences adaptée. Que remarquez-vous ?

Débruitage

4/ Construisez une fenêtre de Hamming de 256 points. Tracez cette fonction dans une nouvelle figure.

5/ Complétez le script SoustractionSpe.m pour appliquer la soustraction spectrale présentée en introduction en parallèle sur les deux blocs. L'équation 2 est déjà implémentée. Faites le même travail pour les équations 3 et 4.

6/ Ecrivez le signal débruité dans un nouveau fichier. Commentez la qualité du débruitage. Vous obtiendrez très probablement un arrière son aigu ressemblant à un bruit de tuyauterie. Modifiez les valeurs de α et β et concluez sur leurs intérêts.