L3 MIA / Systèmes de Communications

TP codage de source

Ce TP utilise scilab, un logiciel libre de simulation numérique (voir `http ://www.scilab.org`). Pour le lancer, sous linux, placez-vous, dans une fenêtre terminal, dans votre répertoire de travail et tapez `"scilab &"`. On peut soit taper directement les instructions dans la fenêtre de commande qui apparaît, soit exécuter un script stocké dans un fichier d'extension .sce. Pour cela, ouvrez ce script dans l'éditeur intégré et faites *exécuter / charger dans scilab*. On peut afficher l'aide en ligne sur une fonction par `help` suivi du nom de la fonction.

Téléchargez `http://www.math-info.univ-paris5.fr/~mahe/TP_ComNum/TP.zip` dans votre répertoire de travail, puis dézippez-le (`unzip TP.zip`).

Pour écouter un son représenté par le vecteur x des échantillons, la commande fournie par la toolbox *boombox.sci* est `soundsc(x,Fe)`, où Fe est la fréquence d'échantillonnage, qui sera fixée à 8000 (Hz) ici.

1 Codage par modélisation AR de la source

Un signal vocal s peut être découpé en tranches de 10 à 30 ms et représenté, sur chaque tranche, par un modèle dit auto-régressif d'ordre p :

$$s(n) = \sigma_e e(n) - \sum_{i=1}^{p} a_i s(n-i)$$

C'est-à-dire que chaque échantillon $s(n)$ est une combinaison linéaire des précédents, plus un terme d'innovation $\sigma_e e(n)$, tel que la puissance de e vaut 1. Nous allons nous appuyer sur cette redondance pour comprimer presque sans perte le signal s. Au lieu de transmettre les échantillons $s(n)$ quantifiés, le codeur transmet pour chaque tranche de 20 ms les coefficients a_1 à a_{10}, σ_e et la suite des $e(n)$ (240 échantillons pour un échantillonnage à 8 kHz)

Ouvrir et lancer *test_codecAR.sce*. Ce programme code un signal sonore s en un flux *coded* et décode celui-ci en un signal sonore s_{codec}. Il affiche s et e. Enfin il quantifie s sur 12 bits, ce qui donne s_q.

Ecoutez et comparez s, s_{codec} et s_q. Baissez le nombre de bits de quantification de s_q jusqu'à la limite de perception du bruit de quantification.

A présent nous allons quantifier e sur un nombre de bit minimal. Dans la fonction *cod_AR*, remplacez :

- l'avant-dernier paramètre par la valeur absolue maximale de e (vous pouvez prendre une valeur légèrement inférieure si peu d'échantillons la dépassent)
- le dernier paramètre par le nombre de bits de quantification de e (10 par exemple).

Lancez le programme, écoutez et comparez s, s_{codec} et s_q. Baissez le nombre de bits de quantification de e jusqu'à la limite de perception du bruit de quantification. Conclusion ?

Si on alloue 100 bits par trame aux coefficients a_i et à σ_e, quel est le gain de codage ?

2 Codage perceptif

Ouvrez le fichier *test_percept.sce*. Ce programme crée un vecteur s qui contient les échantillons d'un fichier son (que vous pouvez choisir à votre guise) de fréquence d'échantillonnage 8 kHz, puis lance une fonction de visualisation du seuil de masquage. Lancez le programme. La fonction *visu_mask* découpe le signal en blocs de 32 ms (256 échantillons) et, pour chacun, affiche le spectre (en noir) et le seuil de masquage (en bleu). Le 2e paramètre de la fonction règle le délai entre 2 figures successives.

Nous allons mettre en oeuvre un codeur perceptif sur le principe suivant : nous transmettrons le signal par blocs de 256 échantillons, en codant la représentation spectrale de chaque bloc ; comme les parties du spectre sous le seuil de masquage ne sont pas audibles, seules seront transmises les parties du spectre au-dessus du seuil de masquage. D'après la visualisation précédente, à quel taux de compression approximatif peut-on s'attendre ?

Mettez en commentaire l'instruction de visualisation et décommentez les 3 lignes suivantes. La fonction *cod_percept*, pour chaque bloc de 256 échantillons,

- calcule le seuil de masquage ;
- génère le spectre échantillonné (256 échantillons fréquentiels). On ne retient que les 129 premiers des 256 échantillons du spectre, car la 2e moitié du spectre peut se déduire de l'autre par conjugaison (NB : mais ce sont des échantillons complexes sauf deux, ce qui revient à 256 valeurs réelles)
- transmet une trame composée de 2 parties : un vecteur de 129 bits, dans lequel chaque i-ème bit indique si le i-ème échantillon du spectre est transmis ; le vecteur des coefficients conservés. Les trames sont donc de longueur variable, mais le 1er vecteur permet au décodeur de connaître la longueur de chacune.

Relancez le programme avec différents fichiers sons. A chaque fois, calculez le taux de compression, en exploitant le fait que chaque trame a pour longueur 129 + le nombre d'échantillons du spectre transmis.