Analyse de Fourier
Vocoder de Phase

C. Dossal

Novembre 2009

1 Introduction

Sous matlab on peut lire un fichier .wav et le transformer en vecteur avec la commande wavread

```
>> [Y,Fs,B]=wavread('FichierSon.wav');
```

Y est le vecteur associé au son, Fs est la fréquence d'échantillonage exprimée en Hertz, B est le nombre de bits sur lequel sont codés les éléments de Y.
La commande wavwrite permet de construire un fichier wav à partir d'un vecteur Y, d'un fréquence Fs et d'un nombre de bits B.

```
>> wavwrite(Y,Fs,B,'FichierSonReconstruit.wav');
```

Si on souhaite ralentir un son, une approche naive consiste à lire le vecteur plus lentement. Idem pour accelerer un son :

```
>> [Y,Fs,B]=wavread('FichierSon.wav');
>> wavwrite(Y,Fs/2,B,'FichierSonreconstruit.wav');
```

Cette modification de la vitesse de lecture induit une modification du contenu fréquentiel du son. Ainsi, si le son est lu deux fois moins vite, il paraitra deux fois plus grave.
Il existe également dans la version windows de matlab une fonction wavplay qui permet d'écouter directement le vecteur sans l'intermédiare d'un fichier .wav.
Le vocoder de phase est un outil qui permet de modifier la longueur du son sans altérer le contenu fréquentiel.

2 Contenu du vocoder

Pour construire un vocoder de phase
- On fenêtre le signal en trames.
- On stocke dans deux tableaux, le module et la phase des fft des trames. Pour dilater ou comprimer un son, on interpole séparérement le module et la phase sur chaque nouvelle trame.
- On reconstruit le signal à partir des nouveaux tableaux de module et de phase.

Le vocoder que nous proposons de construire est une fonction vocoder qui fait ainsi appel à 4 sous programmes :
- Analyse.m qui calcule le spectrogramme et la phase d'un signal S.

- Synthese.m qui reconstruit un signal Srec à partir d'un spectrogramme et d'un tableau de phase.
- InterpSpec.m qui interpole le spectrogramme initial.
- InterpPhase.m qui interpole la phase.

Dans le programme vocoder, on choisira le signal et le facteur de compression ou de dilatation. vocoder crée un fichier .wav directement lisible.
On prendra soin, dans tous les programmes d'utiliser k comme indice de boucle et non i car i désignera le nombre complexe tel que $i^2 = -1$.

3 Les programmes

3.1 L'analyse du son

Ecrire un programme Analyse.m qui prend pour entrée un vecteur colonne S et un entier N et qui renvoie, deux tableaux Spec et Phase. La première ligne du programme est

```
function [Spec,Phase]=Analyse(S,N)
```

Dans l'analyse comme dans la synthèse on utilisera des fenêtres de taille N, décalées de $N/8$. En chaque point passeront ainsi 8 fenêtres.
On tronquera le signal à un nombre entier de fenêtres, en pratique on prendra souvent $N = 1024$.
On pourra utiliser les notations suivantes

- NS taille de S.
- Nf nombre de fenêtres d'analyse.
- H fenêtre de Hanning de taille N.

Remarques :

- $Nf = floor(8 * NS/N) - 7$.
- La fenêtre d'indice k commence à l'indice $1 + (k-1)N/8$ et se termine à l'indice $(k-1)N/8 + N$.
- Ce programme peut être écrit proprement en moins de 15 lignes de matlab.

3.2 La synthèse du son

Ecrire un programme Synthese.m qui prend en entrée deux tableaux, un de module *Spec* et un de phase *Phase* et qui renvoie un vecteur *Srec*.
La première ligne du programme est

```
function Srec=Synthese(Spec,Phase)
```

Remarques

- Il est plus facile de partir d'un vecteur *Srec* nul et d'y ajouter chaque fenêtre que de parcourir les points un à un et de sommer les contributions des 8 fenêtres associées au point.
- On n'oubliera pas de fenêtrer avec une fenêtre de Hanning, les trames reconstruites.
- Ce programme peut également être écrit en moins de 15 lignes.

3.3 Interpolation du spectre

Ecrire un progamme InterpSpec qui prend en entrée, un spectrogramme *Spec* et un vecteur colonne T contenant les "indices réels" des nouvelles fenêtres à construire et qui renvoie le tableau *Spec*2 du spectrogamme interpolé. Les valeurs de T doivent varier entre 1 et $Nf - 1$.
La première ligne du programme est

```
>> Spec2=InterpSpec(Spec,T);
```

Remarques :
- Le nombre de fenêtres de $Spec2$ est égal à la longueur de T
- La colonne d'indice k de $Spec2$ correspond à la colonne d'indice réel $T(k)$ de $Spec$. Elle se calcule à partir des colonnes de $Spec$ d'indices $floor(T(k))$ et $floor(T(k)) + 1$.

3.4 Calcul de la phase

Ecrire un programme InterpPhase qui prend en entrée un tableau de phase $Phase$ et un vecteur T contenant les indices "réels" des nouvelles fenêtre et qui renvoie le tableau de phase $Phase2$ associé à ce jeu de fenêtre.
La première ligne du programme est

```
>> Phase2=InterpPhase(Phase,T);
```

Le calcul de la phase est le point le plus délicat du vocoder de phase. Pour que le son reconstruit soit le plus fidèle possible au son original, la phase associée à chaque fréquence doit varier à la même vitesse que sur le son original.
Comme dans le programme précédent, la colonne d'indice k de $Phase2$ correspond à la colonne d'indice réel $T(k)$ dans le tableau $Phase$. Cette colonne $Phase2(:, k)$ doit être construite de telle sorte que la variation de phase locale entre les deux tableaux soit proche :

$$Phase2(:, k) - Phase2(:, k-1) = Phase(:, floor(T(k))) - Phase(:, floor(T(k)) - 1) \qquad (1)$$

3.5 Le Vocoder

On appellera par exemple Vocoder.m le programme principal. Il prend en entrée un fichier .wav et un facteur de dilatation $alpha$.

- On analyse le son grâce au programme Analyse.m .
- On crée un vecteur T qui contiendra les "indices réels" des fenêtres associées au signal reconstruit.
  ```
  >> T=[1:alpha:Nf-1];
  ```
- On interpole le spectre et on calcule la phase avec InterpSpec.m et InterpPhase.m .
- On reconstruit le son avec Synthese.m .
- Grâce à la commande wavwrite on crée un fichier .wav associé au vecteur reconstruit.