Représentations Temps-Fréquence : travaux pratiques

Compte-rendu à transmettre à `Sebastien.Bourguignon@ec-nantes.fr`, **au format pdf** , *au plus tard 10 jours après la séance.*

1 Boîte à outils tftb : mode d'emploi

Pour calculer les différentes représentations temps-fréquence, on utilisera la boîte à outils **tftb** (`http://tftb.nongnu.org`) :

- Récupérer le fichier **tftb-0.2.zip** sur le serveur pédagogique. Un guide d'utilisation est également disponible.
- Le décompresser à la racine de votre répertoire de travail.
- Pour pouvoir exécuter les fonctions contenues dans ce dossier, déclarer le chemin d'accès à Matlab au début de votre script :
  `addpath(genpath('`*votre_chemin*`/tftb-0.2'));`
- Récupérer également la fonction `ajoute_bruit.m` et les fonctions de type `tftb_*.m` sur le serveur et les placer au bon endroit.

Les fonctions nécessaires pour ce TP s'utilisent de la façon suivante :

- `[tfrx,T,F] = tftb_spectrogram(x,Nf,h);` calcule le **spectrogramme** du signal `x` défini aux instants `t`, caractérisé par la fenêtre `h` et où les transformées de Fourier sur chaque séquence sont calculées sur `Nf` points ;
- `[tfrx,T,F] = tftb_wvd(x,Nf);` calcule la **transformée de Wigner-Ville** de `x` ;
- `[tfrx,T,F] = tftb_pwvd(x,Nf,h);` calcule la **transformée de pseudo-Wigner-Ville** de `x`, où `h` représentent la fenêtre temporelle ;
- `[tfrx,T,F] = tftb_spwvd(x,Nf,g,h);` calcule la **transformée de pseudo-Wigner-Ville lissée** de `xa`, où les fréquences sont calculées sur `Nf` points, et où `g` et `h` représentent les fenêtres temporelle et spectrale.

Les variables en sortie contiennent la représentation temps-fréquence `tfrx` et les axes temporel et fréquentiel `T` et `F`. On visualise le résultat par l'affichage de l'image :
`imagesc(T,F,tfrx); axis xy`.

Remarque : l'ensemble des représentations temps-fréquence sont renvoyées pour l'intervalle de fréquences normalisées $[0, 1/2]$.

2 Représentations temps-fréquence de signaux simulés

2.1 Génération de signaux

Synthétiser les signaux suivants, avec $N = 256$ et $n = 0, \ldots, N-1$:

— x_1 et x_2, définis à partir de leur phase instantanée sous la forme :

$$x_1[n] = \cos(2\pi\phi_1[n]), \qquad \text{avec} \qquad \phi_1[n] = 0.25n + 5\cos(2\pi n/N)$$
$$x_2[n] = \cos(2\pi\phi_2[n]), \qquad \text{avec} \qquad \phi_2[n] = 0.25n + 0.1n^2/N.$$

Pour ces deux signaux *mono-composantes*, représenter leur allure temporelle, leur phase instantanée et leur fréquence instantanée en fonction du temps [1]. Pour rappel, $f_i(t) = d\phi_i(t)/dt$.

— x_3, somme de quatre atomes gaussiens de taille $N_h = 51$, sous la forme :

```
T1 = 15; T2 = 85; T3 = 180;
Nh = 51; th = (0:Nh-1); h = gausswin(Nh)';
x3 = zeros(1,N);
x3(T1:T1+Nh-1) = h.*cos(2*pi*0.25*th);
x3(T2:T2+Nh-1) = h.*cos(2*pi*0.15*th);
x3(T2:T2+Nh-1) = x3(T2:T2+Nh-1) + h.*cos(2*pi*0.35*th);
x3(T3:T3+Nh-1) = h.*cos(2*pi*0.3*th);
```

Représenter l'allure de ce signal en fonction du temps.

2.2 Représentations temps-fréquence

Pour chacun de ces trois signaux, calculer :

— le spectrogramme, en utilisant une fenêtre `h` de Hamming [2] de longueur `Nh` = 17, 33, 65 et 129. On affichera les quatre résultats sur la même figure (`subplot`) ;

— la transformée de Wigner-Ville ;

— la transformée de pseudo-Wigner-Ville, en utilisant une fenêtre `h` de Kaiser [3] avec `Nh` = 63 ;

— la transformée de pseudo-Wigner-Ville lissée, en utilisant des fenêtres `g` et `h` de Kaiser de longueur `Ng` =33 et `Nh` =63, puis `Ng` = 15 et `Nh` = 63. On affichera les transformées de Wigner-Ville sur la même figure.

En analysant les résultats sur l'ensemble des signaux, commenter les performances de ces représentations, en particulier en fonction de leurs paramètres respectifs.

2.3 Influence du bruit

Perturber maintenant les trois signaux avec un bruit additif gaussien, pour un rapport signal sur bruit de 10 dB (cf. fonction `ajoute_bruit.m`). Observer les signaux temporels et les différentes représentations précédentes. Conclure sur leur robustesse au bruit.

1. On pourra utiliser la fonction `subplot` pour l'affichage.
2. Fonction `h = hamming(Nh);`
3. Fonction `h = kaiser(Nh);`

3 Détection et reconstruction d'une partition musicale

On se propose maintenant d'exploiter les représentations temps-fréquence (RTF) pour la reconstruction automatique d'une partition musicale. *L'exemple est ici simplifié et restreint à l'identification d'une seule note à chaque instant.*

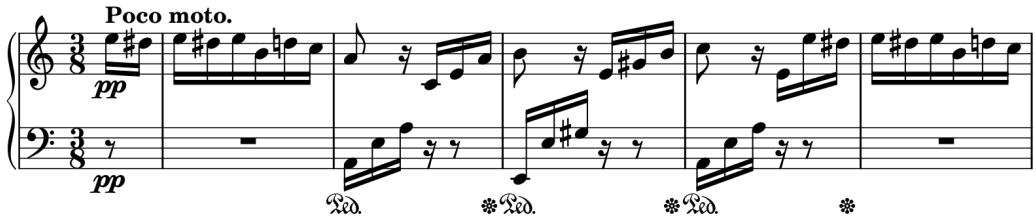

FIGURE 1 – Extrait de partition musicale. Sur chacune des deux lignes, la hauteur représente la note jouée. L'axe horizontal représente le temps, le motif associé à chaque note représente sa durée.

3.1 Principe de la méthode

On envisage une méthode détaillée en les étapes suivantes.

1. **On recherche les changements de contenu fréquentiel au cours du signal, en mesurant un indice de stationnarité :**
   — pour chaque instant t, on compare les RTF du signal sur une plage temporelle antérieure à t (`R(t-`τ`:t,f)`) et postérieure à t (`R(t:t+`τ`,f)`). On définit alors un indice de stationnarité $I(t)$ à partir d'une mesure de ressemblance entre ces deux *imagettes*, la distance de Kolmogorov :

$$I(t) = \sum_u \sum_f \left| \widetilde{R}_1(u,f) - \widetilde{R}_2(u,f) \right| \text{ où } \widetilde{R}_i(u,f) = \frac{R_i(u,f)}{\sum_{u,f} \left| R_i(u,f) \right|};$$

   — les ruptures de stationnarité sont alors détectées par les maxima locaux [4] dans l'indice de stationnarité $I(t)$, dont l'amplitude est supérieure à un seuil donné.
2. **Dans chaque plage de stationnarité, on calcule la fréquence pour laquelle la transformée de Fourier est maximale en module** – on suppose ici que l'on cherche uniquement la composante harmonique de plus grande amplitude.
3. **On associe à cette fréquence la note la plus proche** dans le tableau de référence :

Note	do1	reb1	re1	mib1	mi1	fa1	solb1	sol1	lab1	la1	sib1	si1	
Fréquence (Hz)	262	278	294	312	330	350	370	392	416	440	466	494	
Note	do2	reb2	re2	mib2	mi2	fa2	solb2	sol2	lab2	la2	sib2	si2	do3
Fréquence (Hz)	524	556	588	624	660	700	740	784	832	880	932	988	1048

TABLE 1 – Fréquences des notes de la gamme tempérée sur un octave.

Ces valeurs sont fournies dans le fichier `freq_notes.mat`, contenant une variable par note (exemple : la variable `re1` vaut 147). Les variables `tab_freq_noms` et `tab_freq_valeurs` reprennent la liste des noms des notes et leurs fréquences.

4. Fonction `findpeaks.m`

3.2 Travail demandé

1. Charger le fichier `furElise_court.wav`.
2. L'écouter, puis observer et commenter son spectrogramme. *Pour améliorer la lisibilité, on pourra afficher l'amplitude de l'image en échelle logarithmique.*
3. Mettre en œuvre la méthode détaillée au § 3.1. La méthode renverra en sortie la note détectée et sa durée dans chaque plage de stationnarité.
4. Afin d'analyser la qualité de la transcription, resynthétiser le signal musical à partir des seules notes détectées. La fonction `s = genere_morceau(tab_notes,tab_duree,Fe);` génère une succession de signaux sinusoïdaux, de durées `tab_duree` (en s), et échantillonné à la fréquence Fe (ex. `tab_notes = [do1; mi1; sol1]; tab_duree = [1; 0.5; 1];`).
5. Écouter le morceau synthétique. Commenter les performances de la méthode.

BONUS Recommencer l'analyse avec les autres fichiers `.wav` disponibles sur le serveur. Pourquoi est-ce plus difficile sur ces extraits musicaux ?