Transformée en ondelettes discrète : applications

Stage LIESSE / T.I.P.E.

Avant de commencer le TP, il faut :
- effectuer le log-in avec nom d'utilisateur : `tpinvi` et mot de passe : `snhnzSB`
- créer un répertoire de travail ;
- télécharger le fichier supplémentaire d'ici : `http://wp.cagnazzo.mines-telecom/` →`Stage LIESSE/T.I.P.E.`
- extraire tous les fichiers des archives compressés dans le répertoire de travaille.
- ouvrir une fenêtre de terminal et lancer Matlab avec la commande : `/opt/matlab/bin/matlab`
- Ajouter le toolbox Wavelab850 au "path" de Matlab : File → Set Path → **Add with subfolders**

1 Analyse multirésolution

Calculez la réponse impulsionnelle du filtre passe-bas de l'AMR orthogonale de Daubechies d'ordre 4 (c'est-à-dire, avec 4 moments nuls) au moyen de la fonction `qmf = MakeONFilter('Daubechies',8);`

1. Déterminez la réponse fréquentielle de ce filtre en effectuant une TFD sur 256 points : `N=2^8; H=fft(qmf,N);` Représentez graphiquement le module de cette réponse fréquentielle : `nu = (0:N-1)/N; plot(nu,abs(H));` Comment ce graphe est-il modifié quand on augmente ou on diminue l'ordre du filtre ?

2. Créez plusieurs signaux avec la fonction `MakeSignal` : rampe, polynomial par morceau. Tapez `help MakeSignal` pour un guide sur l'utilisation de cette fonction. Effectuez la décomposition 1D en ondelettes de Daubechies-8 périodisées de ces signaux, au moyen de la fonction `FWT_PO`[1]. Qu'observez-vous, en particulier par rapport aux détails et aux discontinuités ? Que se passe-t'il en utilisant des filtres de tailles différentes ?

3. Chargez l'image `lena.pgm` avec les fonctions `imread` et transformez-la en format double avec la commande `double` (nécessaire pour manipuler l'image). Affichez l'image avec les commandes `image` et `colormap(gray(256))`, ce dernier assurant une correcte visualisation.

4. Réalisez les décompositions 2D en ondelettes de l'image lena avec la fonction `FWT2_PO` sur 1 niveau puis sur 2 niveaux et visualisez-les avec `image`, avec `imagesc`, et avec `wt_view`. Qu'observez-vous et quelles conclusions concernant la compression pouvez-vous en tirer ?

5. Reprenez les questions précédentes, en générant des ondelettes biorthogonales à l'aide de la fonction `[qmf,dqmf] = MakeBSFilter(Type,Par)`, avec `Type ='Villasenor'` et `Par=1`, ce qui correspond aux ondelettes 9/7, et en réalisant la décomposition multirésolution au moyen des fonctions `FWT_SBS` et `FWT2_SBS`. Que constatez-vous par rapport aux problèmes d'extension de l'image (périodique/symétrique) ?

Le corrigé de cette partie est disponible dans le fichier `tp_amr.m`

1. Soit $N = 2^K$ la durée du signal, et soit M le nombre de niveaux de décomposition souhaités. Le paramètre `L` de la fonction `FWT_PO` doit alors être égal à $K - M$.

2 Allocation de débit et quantification

On fixe pour la suite une décomposition Daubechies-9/7 et un nombre de niveaux de décomposition J = 4.

1. Utilisez la fonction `bd = getsb(wc,i,J)`, capable d'isoler la i-ème sous-bande de la décomposition multirésolution de J niveaux pour visualiser chaque sous-bande. Calculez la variance des sous-bandes résultantes. Affichez aussi les histogrammes de ces sous-bandes au moyen de la fonction `hist` : Quelles observations pouvez-vous faire ?

2. On peut allouer d'une manière optimale le débit en fonction des variances des sous-bandes en utilisant la relation suivante :

$$b_i = \bar{b} + \frac{1}{2}\log_2\left[\sigma_i^2 / \prod_{j=1}^{I} {\sigma_j^2}^{\frac{N_j}{N}}\right], \qquad \forall i \in \{1, 2, \ldots, I\}$$

   où b_i représente le nombre de bits/pixel dans la sous-bande i, I est le nombre total de sous- bandes,$\bar{b}$ est le débit moyen que l'on s'impose, σ_i^2 la variance de la sous-bande i, N_i le nombre de points de cette même sous-bande et N le nombre total de points dans l'image. Utilisez la fonction `ratealloc` pour calculer le débit de codage de l'image `lena` pour un débit global de $\bar{b} = 1$ bit/pixel.

3. Utilisez la fonction `wcq = quantsb(wc,J,b)` pour réaliser la quantification optimale des sous-bandes de débits b. Quantifiez les coefficients de la transformée au débit global fixé de $\bar{b} = 1$ bit/pixel. Affichez les histogrammes des sous-bandes quantifiées.

4. Réalisez la synthèse de l'image à partir des coefficients quantifiés :
   `imq=IWT2_SBS(wt2d_q, imgLogSize-nDecLev,qmf,dqmf);` Visualisez l'image reconstruite et comparez-la à l'image originale en calculant le PSNR défini, pour deux images a et b, par :

$$\mathrm{PSNR}(a,b) = 10\log_{10}\left(\frac{255^2}{\frac{1}{N}\sum_{i=1}^{N}(a_i - b_i)^2}\right)$$

5. Calculez la courbe débit-PSNR pour le codage de l'image `lena`, avec un débit qui varie entre 0.2 et 2 bpp.

6. La fonction `entropyQ` permet d'estimer le taux de codage qu'on peut atteindre avec un codeur à longueur variable (codage sans pertes). Calculez l'entropie (donc le taux de codage estimé, en bit par pixel) de l'image originale : `entropyQ(img)`
   Calculez l'entropie des coefficients quantifiés : `wtqEntropy(wt2d,nDecLev,bi)`

Le corrigé de cette partie est disponible dans le fichier `tp_compression.m`

3 Mouvement Brownien Fractionnaire

3.1 Rappels

Définition du signal de mouvement Brownien fractionnaire :
Processus Gaussien, centrée, réel, non stationnaire, de covariance :

$$\mathrm{E}\{B_H(t)B_H(s)\} = \frac{V_H}{2}\left[|t|^{2H} + |s|^{2H} - |t-s|^{2H}\right], \qquad H \in]0,1].$$

Le mouvement Brownien est un cas particulier, pour $H = 1/2$. H est appelé indice de Hurst.
Ce modèle est non-stationnaire. En particulier, la variance de $B_H(t)$ varie avec le temps :

$$\mathrm{var}\{B_H(t)\} = V_H|t|^{2H}$$

Ce processus est néanmoins auto-similaire : $B_H(at)$ a la même distribution de probabilité de $a^H B_H(t)$. En plus, ses accroissements sont stationnaires :

$$\Delta B_H(t,\tau) = B_H(t+\tau) - B_H(t) \tag{1}$$

$$\mathrm{var}\{\Delta B_H(t,\tau)\} = \frac{V_H}{2}|\tau|^{2H} \tag{2}$$

L'analyse en ondelettes du mBf présente des aspects intéressants :
- La non-stationnarité demande une analyse dépendante du temps ;
- L'auto-similarité demande une analyse dépendante de l'échelle.

En particulier, si on appelle $c_j[\cdot]$ les coefficients de détails de niveau j :

$$d_j[k] = \frac{1}{2^j}\int_{-\infty}^{+\infty} B_H(t)\Psi\left(\frac{t}{2^j} - k\right)dt, \tag{3}$$

$$c_j[k] = \frac{1}{2^j}\int_{-\infty}^{+\infty} B_H(t)\Phi\left(\frac{t}{2^j} - k\right)dt, \tag{4}$$

il est possible de montrer que les coefficients de détails sont stationnaires, avec variance :

$$\mathrm{var}\{c_j(\cdot)\} = \mathrm{const}(2^j)^{2H+1} \tag{5}$$

3.2 Exemples de signal mBf

1. Charger les données avec la commande :`load brownian`. Dans la variable `mBf` vous avez 30 réalisation de quatre processus de mouvement Brownien fractionnaire, avec quatre valeurs de H différents.
   Avec `x = mBf(:,20,1)` on peut accéder à la 20ème réalisation du premier processus.
   Afficher une réalisation de chacun des 4 processus. Utiliser le zoom pour regardes les détails des signaux. Est-ce que cela permet d'ordonner les processus par H croissant ?

3.3 Estimation de l'indice H par TO

1. Calculer la décomposition en ondelettes d'une réalisation d'un des quatre signaux. Utiliser le filtre orthogonale de Daubechies de longueur 8 : `qmf = MakeONFilter('Daubechies',8);`

2. Sélectionner un niveau de décomposition du signal transformée. Estimer la variance du signal $c_j[\cdot]$.

3. Pour l'estimation de la variance d'un signal Gaussien, on peut exploiter le fait que lorsque Z a une distribution normale, la médiane de $|Z|$ est égal à $0.6745\sigma_Z$

4. En considérant l'équation (5), comment peut-on utiliser les estimation des variances pour estimer H ?

5. Sélectionner toute réalisation d'un des signaux, par exemple le premier.
   `signalN = 1; mbf =mBf(:,:,signalN);`

6. Calculer toutes TOD des réalisations.
```
wt=zeros(size(x));
for m=1:M,
wt(:,m) = FWT_PO(x(:,m), log2(N)-nDec, qmf);
end
```

7. Pour chaque niveau, estimer les variances des différentes réalisations :
```
for lev =1:nDec,
start = N/(2^lev)+1;
stop = N/(2^(lev-1));
det = wt(start:stop,:);
vEst(lev,:)= median(abs(det))/0.6745;
end
```
`vEst` est la matrice des estimations des déviations standards. Une colonne correspond à une réalisation.

8. Calculer le $\log_2$ des variances :
$$\log_2 \operatorname{var}\{c_j(\cdot)\} = (2H+1)j + \text{const}$$
Calculer $2H+1$ par régression linéaire : pour chaque réalisation,
```
p = polyfit(levs,log2(v),1);
Hest = (p(1)-1)/2;
```
où `v` est le vecteurs des variances estimées pour une réalisation, et `levs` est le vecteur des j. Pour améliorer la robustesse, il convient de ne pas prendre en compte le niveau de résolution le plus fin.

Le corrigé de cette partie du TP est disponible dans le fichier `tp_fbm.m`