Transformée Discrète en Ondelettes : travaux pratiques

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

Rappels et mise en œuvre avec la boîte à outils `Wavelab`

Soit un signal $x[n]$ de N points avec $N = 2^P$. Sa transformée discrète en ondelettes (discrete wavelet transform, DWT) jusqu'à l'échelle J (avec $J \leqslant P$) est représentée par :

- les coefficients de détail à l'échelle 1 ($N/2$ coefficients) ;
- les coefficients de détail à l'échelle 2 ($N/4$ coefficients) ;
- ...
- les coefficients de détail à l'échelle J ($N/2^J$ coefficients) ;
- les coefficients d'approximation à l'échelle J ($N/2^J$ coefficients).

Les coefficients de l'approximation à l'échelle j s'obtiennent par filtrage des coefficients de l'approximation à l'échelle $j-1$ (filtre h, passe-bas). Les coefficients de détail à l'échelle j s'obtiennent par filtrage des coefficients de l'approximation à l'échelle $j-1$ (filtre g, passe-haut). On considère ici des transformées orthogonales (cf. cours) telles que $g[n] = (-1)^{1-n}h[1-n]$. **La transformée est donc complètement définie par la séquence génératrice h et l'échelle finale J.**

Télécharger la boîte à outils `Wavelab` sur le serveur pédagogique [1]. On n'utilisera ici que les fonctions suivantes :

— `h = MakeONFilter('Haar'); h = MakeONFilter('Daubechies',n);`
calcule le filtre échelle pour une ondelette de Haar ou de Daubechies d'ordre n

— `DWT_x = FWT_PO(x,L,h);` calcule la transformée discrète en ondelettes, pour les signaux 1D. **Attention : $L = P - J$, où J est la dernière échelle souhaitée.**

— `x = IWT_PO(DWT_x,L,h);` calcule la transformée inverse 1D

— `DWT_I = FWT2_PO(I,L,h);` calcule la transformée discrète en ondelettes 2D pour les images

— `I = IWT2_PO(DWT_I,L,h);` calcule la transformée inverse 2D.

Télécharger également deux fonctions d'affichage sur le serveur pédagogique :

— `plot_dwt(x,DWT_x,J);` affiche le signal et sa DWT, en représentant sur plusieurs lignes les différents coefficients de détail $\boldsymbol{d}_1, \ \ldots \boldsymbol{d}_J$ et d'approximation $\boldsymbol{a}_J$ selon la syntaxe (1) ;

— `im_dwt2(DWT_I,J);` représente sous forme d'image la DWT contenue dans la matrice `DWT_I`.

1. également récupérable ici : `http://www-stat.stanford.edu/~wavelab`

1 Tracé d'ondelettes et de fonctions échelles par DWT inverse

La DWT, qui se construit à partir de la séquence génératice h, définit implicitement une fonction ondelette $\psi(t)$ et une fonction échelle $\varphi(t)$, telles que :

$$d_{j,n} = \langle x, \psi_{j,n} \rangle, \quad \forall j \leqslant J, \quad \forall n \in \mathbb{Z}$$
$$a_{J,n} = \langle x, \varphi_{J,n} \rangle, \quad \forall n \in \mathbb{Z}$$

Puisque l'on considère des ondelettes orthogonales [2], on a, pour tout signal $x(t)$:

$$x(t) = \sum_{j=1}^{J} \underbrace{\sum_{n \in \mathbb{Z}} \overbrace{\langle x, \psi_{j,n} \rangle}^{d_{j,n}} \psi_{j,n}(t)}_{\text{signal de détail à l'échelle } j} + \underbrace{\sum_{n \in \mathbb{Z}} \overbrace{\langle x, \varphi_{J,n} \rangle}^{a_{J,n}} \varphi_{J,n}(t)}_{\text{approximation du signal à l'échelle } J} .$$

Par conséquent, pour $x(t) = \psi_{j_0,n_0}(t)$, la DWT de x est nulle partout sauf en j_0, n_0 et ψ_{j_0,n_0} peut être obtenu par transformée inverse de cette DWT.

On considère un signal de $N = 1024$ points (correspondant à l'échelle 0).

Sous forme informatique (c'est le cas de la boîte à outils WaveLab), on représente en général la DWT par le vecteur de N points contenant les coefficients dans l'ordre suivant :

$$\mathrm{DWT}(\boldsymbol{x}) = [\ \underbrace{\boldsymbol{a}_J}_{\frac{N}{2^J}\text{coefs}}\ ,\ \underbrace{\boldsymbol{d}_J}_{\frac{N}{2^J}\text{coefs}}\ ,\ \underbrace{\boldsymbol{d}_{J-1}}_{\frac{N}{2^{J-1}}\text{coefs}}\ , \ldots ,\ \underbrace{\boldsymbol{d}_1}_{\frac{N}{2}\text{coefs}}\ ]. \tag{1}$$

1. **À quel indice** dans la représentation vectorielle (1) correspond le coefficient d'approximation $a_J[k]\ \forall k$? À quel indice correspond le coefficient de détail $d_j[k]\ \forall j, k$?
2. Construire le vecteur `DWT x` contenant seulement un coefficient non-nul, de détail, à la plus grande échelle, et de décalage temporel le situant approximativement au milieu de l'axe temporel.
3. En déduire et tracer la fonction ondelette (discrétisée à l'échelle 0), correspondant à la transformée de Haar ?
4. **De même, tracer** les fonctions échelles (à l'échelle 0) correspondantes.
5. Pour ces deux signaux (questions 3 et 4), calculer la transformée en ondelettes et la tracer (fonction `plot_dwt.m`), vérifier le résultat.
6. Répéter la procédure pour les ondelettes de Daubechies d'ordre 4 et 8.
7. **Commenter** la forme des fonctions obtenues.

2. On a donc $\langle \psi_{j,n}, \psi_{j',n'} \rangle = \delta[j - j']\ \delta[n - n']$ et $\langle \psi_{j,n}, \varphi_{j,n} \rangle = 0$

2 ≪ Débruitage ≫ dans l'espace des ondelettes

Pour cette partie, on considérera au choix une ondelette de Haar ou de Daubechies d'ordre 4.

1. **Créer** un signal de $N = 1024$ points dont la DWT jusqu'à l'échelle $J = 7$ contient uniquement quelques coefficients de détail non nuls. Visualiser le signal et sa DWT (fonction `plot_dwt`).
2. **Ajouter du bruit** au signal précédent (on utilisera la fonction `ajoute_bruit` utilisée dans le premier TP, qui renverra le signal bruité et l'écart-type σ_b du bruit). Visualiser le signal bruité et sa DWT et comparer au signal sans bruit.

   Une procédure dite de ≪ débruitage ≫ consiste alors à ne garder que les coefficients significatifs dans la DWT :

   (a) Calculer la DWT jusqu'à l'échelle J.

   (b) Seuiller ses coefficients en ne conservant que ceux dont l'amplitude (**en valeur absolue**) dépasse le seuil α (les autres étant mis à 0).

   (c) Calculer la DWT inverse des coefficients seuillés.

   (d) Mesurer la qualité du signal reconstitué par rapport au signal initial.

3. **Mettre en œuvre et répéter cette procédure** pour plusieurs valeurs du niveau de seuil α et analyser l'erreur de reconstruction commise en fonction de α. Comparer la valeur obtenue pour le seuil optimal à celle proposée par Donoho & Johnstone[3] : $\alpha^* = \sigma_b\sqrt{2\ln N}$.

Application à des signaux plus réalistes :

4. **Générer des signaux** "Blocks" : `[x,N] = makesig('Blocks',N);`
   et "Doppler" : `[x,N] = makesig('Doppler',N);`
5. **Visualiser** ces deux signaux ainsi que leur DWT et commenter. Rajouter du bruit avec un RSB de 20 dB et appliquer la procédure précédente
6. **Comparer les performances** de débruitage obtenues en utilisant une ondelette de Haar et une ondelette de Daubechies d'ordre 4, pour chacun des deux signaux, et commenter.

3. D. L. Donoho and J. M. Johnstone, *Ideal spatial adaptation by wavelet shrinkage*, Biometrika (1994), vol. 81.

3 Compression d'images

Le standard de compression JPEG 2000 exploite la *parcimonie* des coefficients de décomposition en ondelettes des images. On propose ici de mettre en évidence cette propriété sur des images naturelles, puis d'analyser les performances de compression qui en résultent.

1. **Récupérer l'image** `lena.bmp` disponible sur le serveur pédagogique et l'ouvrir sous Matlab : `I = double(imread('lena.bmp'));`. *Vous pouvez aussi prendre toute autre image de votre choix, de taille carrée $N \times N$, où N est une puissance de 2.* On considère une image en niveaux de gris, obtenue éventuellement en ajoutant les trois bandes couleur : `I = sum(I,3);`.
2. **Visualiser l'image :** `imagesc(I); colormap gray; axis square`
3. **Calculer sa transformée en ondelettes 2D** à l'échelle $J = 4$ en utilisant un filtre générateur de Daubechies d'ordre 4 (voir en première page).
4. **Visualiser cette DWT-2D** (voir en première page). *Attention, à des fins de visualisation, chaque sous-bloc carré de la DWT-2D a été normalisé à une échelle différente.*
5. **Analyser la distribution des coefficients :** représenter leur histogramme (`hist`) et tracer les coefficients par ordre décroissant en valeur absolue (`sort`). Commenter cette distribution par rapport à celle des valeurs des pixels de l'image.
6. **Réaliser la procédure de compression :**
   (a) Seuiller les coefficients de la DWT en ne conservant que les $\tau\%$ les plus grands (**en valeur absolue**), les autres étant mis à 0.
   (b) Reconstruire l'image correspondant à cette DWT seuillée.

   La *compression* réside dans l'étape 1, qui signifie bien qu'en pratique, on peut représenter (et donc stocker) l'image en ne conservant que les valeurs et les coordonnées des plus grands coefficients.
7. **Visualiser l'image reconstruite** et l'image d'erreur par rapport à l'image initiale, pour plusieurs valeurs de τ que l'on interprétera en termes de taux de compression [4]. Selon le temps disponible, on pourra, comme précédemment, mesurer l'erreur entre l'image compressée et l'image initiale pour différentes valeurs de τ, et choisir ainsi τ.
8. **Recommencer la procédure** en choisissant une ondelette de Haar. Comparer les résultats obtenus.

4. Pour cela, on peut comparer la place occupée en mémoire par la variable Matlab contenant l'image originale et celle contenant la DWT seuillée, codée en format `sparse`.