# Compression d'images par ondelettes

Benjamin CATINAUD

Année 2017-2018

**Résumé**

Nowadays, million of pictures are sent every minute. Consequently, we have to create algorithms to compress this data. To this end, several solutions exist. The most famous is perhaps the Fast Fourier Transform, use by scientists. However, this transform take care of frequencies in a signal but doesn't take its chronology into account. To adress this issue, I suggest to study the Wavelet Transform. This paper will detail basics of wavelet theory, as well as algorithms to transform and rebuild pictures with wavelets and, finally, discuss their efficiency.

# Introduction

Aujourd'hui où des millions d'images sont envoyées chaque minute, une méthode pour compresser ces données, situées à l'interface ténue entre humain et machine et témoignant des interactions humaines, est par conséquent requise. Cette méthode devra optimiser rapidité, homogenéité, integrité et cohésion de ces informations.

Pour ce faire, plusieurs solutions peuvent être mises en place. Typiquement, les physiciens utilisent dans la plus grande majorité des cas la transformation $FFT$ (pour Fast Fourier Transform).

Cependant, la transformée de Fourier permet de bien distinguer les fréquences d'un signal mais pas leur chronologie [4]. Cette particularité peut être problématique dans certains cas lors de traitements du signal comme le cas de signaux non sinusoïdaux, notamment pour les images. Ainsi, nous choisirons de travailler avec la transformation par ondelettes, fonctions de carré intégrable, à support compact et d'intégrale nulle.

La théorie des ondelettes, mise au point dans les années 1980 par plusieurs chercheurs dont notamment Yves Meyer et Jean Morlet, a connu ainsi un rapide succès que ce soit dans le traitement de signaux sismiques (domaine qui a initié la recherche dans cette théorie) ou dans le traitement des images.

Ce travail va donc s'appuyer sur la théorie des ondelettes afin de répondre au problème de la compression d'images en utilisant à profit cette théorie. Après avoir donné la définition d'une ondelette et avoir défini l'analyse multirésolution, nous détaillerons l'algorithme de la transformation par ondelettes (abrégé dans la suite l'algorithme $FWT$) conçu par Stéphane Mallat. Ensuite, nous discuterons de l'efficacité de cet algorithme du point de vue de la complexité ainsi que les pertes occasionnées.

# I. Une ondelette, c'est quoi ?

**Définition 1 (Ondelette mère et ondelettes filles)**

*Soit* $\psi \in L^2(\mathbb{R},\mathbb{R})$ *à support compact et telle que* $\int_{\mathbb{R}} \psi$ *converge et vaut* 0.

*Soit* $(\psi_{j,k})_{(j,k)\in\mathbb{Z}^2}$ *une famille de fonctions de* $L^2(\mathbb{R},\mathbb{R})$ *définie par :*

$$\forall (j,k) \in \mathbb{Z}^2, \forall x \in \mathbb{R}, \psi_{j,k}(x) = \frac{1}{\sqrt{2^j}}\psi(\frac{2^j x - k}{2^j})$$

*On dit alors que* $\psi$ *est l'ondelette mère [4] et que la famille* $(\psi_{j,k})$ *issue de translations et de dilatations de l'ondelette mère forme la famille des ondelettes filles.*

**Exemple** Ondelette de Haar

Soit $\psi$ la fonction définie par : $\psi = \mathbb{1}_{[0;\frac{1}{2}[} - \mathbb{1}_{[\frac{1}{2};1[}$. [1]

$\psi$ étant constante par morceaux, $\int_{\mathbb{R}} \psi$ converge et vaut 0.

De plus, son support est clairement compact.

La famille orthonormale associée à l'ondelette mère $\psi$ est alors :

$$(\psi_{j,k})_{(j,k)\in\mathbb{Z}^2} \text{ où } \forall j,k \in \mathbb{Z}, \forall x \in \mathbb{R}, \psi_{j,k}(x) = \frac{1}{\sqrt{2^j}}\psi(\frac{x-2^j k}{2^j})$$

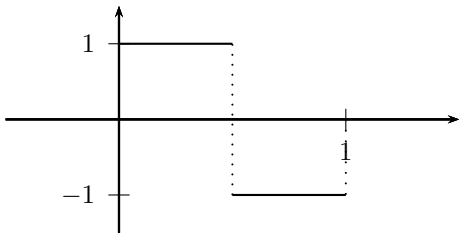


FIGURE 1 – Courbe représentative de $\psi$

# II. Analyse multirésolution

**Définition 2 (Espace d'approximation)**

*Soit* $j \in \mathbb{Z}$. *On appelle espace d'aproximation à l'échelle* $2^j$ *l'espace vectoriel des fonctions de* $L^2(\mathbb{R},\mathbb{R})$ *constantes sur les intervalles de la forme* $[2^j k; 2^j(k+1)[,\ k \in \mathbb{Z}$ *[1]. On le note* $V_j$.

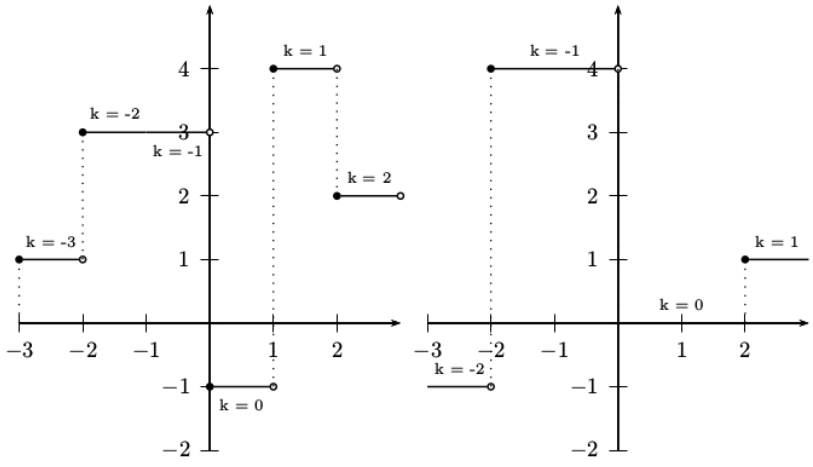


FIGURE 2 – Exemple graphique d'une fonction de $V_0$ (à gauche) et de $V_1$ (à droite)

**Proposition 1**

*Les espaces $(V_j)_{j\in\mathbb{Z}}$ vérifient les propriétés suivantes [1] :*

$(i)$ $\forall j \in \mathbb{Z}, V_{j+1} \subseteq V_j$

$(ii)$ $\bigcup_{j\in\mathbb{Z}} V_j$ *est dense dans* $L^2(\mathbb{R},\mathbb{R})$

**Proposition 2 (Caractérisation des espaces d'approximation)**

*On a :*

$$\forall j \in \mathbb{Z}, V_j \;=\; \left\langle \left\{ \phi_{j,k} = \tfrac{1}{\sqrt{2^j}} \mathbb{1}_{[2^j k; 2^j(k+1)[},\ k \in \mathbb{Z} \right\} \right\rangle$$

*La famille $(\phi_{j,k})_{(j,k)\in\mathbb{Z}^2}$ est ainsi une famille d'ondelettes filles issue de l'ondelette mère $\phi = \mathbb{1}_{[0;1[}$.*

L'idée de la transformation par ondelettes est ainsi de calculer les projections d'une fonction $f \in L^2(\mathbb{R},\mathbb{R})$ sur les espaces $V_j$ pour un $j \in \mathbb{Z}$ donné. Plus le $j$ diminue, plus l'approximation ainsi faite est précise.

**Définition 3 (Espace des détails)**

*On définit la notion d'espace de détails $W_j$, pour $j \in \mathbb{Z}$, tel que*

$V_{j-1} = V_j \overset{\perp}{\bigoplus} W_j$

*Ainsi, par définition, la connaissance de $W_j$ et $V_j$ permettent d'obtenir $V_{j-1}$.*

**Proposition 3 (Caractérisation des espaces de détails)**

*On a :*

$$\forall j \in \mathbb{Z}, W_j \;=\; \left\langle \left\{ \psi_{j,k} \;=\; \tfrac{1}{\sqrt{2^j}} (\mathbb{1}_{[2^{j-1}k; 2^{j-1}(k+1)[} - \mathbb{1}_{[2^{j-1}(k+1); 2^{j-1}(k+2)[}),\ k \in \mathbb{Z} \right\} \right\rangle$$

**Remarque 1**

*On remarque une relation de récurrence entre les ondelettes $\psi_{j,k}$ et $\phi_{j,k}$ pour tout $k, j \in \mathbb{Z}$ :*

$$\phi_{j,k} = \tfrac{1}{\sqrt{2}}(\phi_{j-1,2k} + \phi_{j-1,2k+1})$$
$$\psi_{j,k} = \tfrac{1}{\sqrt{2}}(\phi_{j-1,2k} - \phi_{j-1,2k+1})$$

Cette remarque nous amène ainsi à l'algorithme de Mallat qui permet la transformation par ondelettes d'une image.

# III. Algorithme de transformation par ondelettes ou Algorithme de Mallat

Dans toute la suite, pour illustrer les propos, nous utiliserons l'image suivante


FIGURE 3 – Image originale

**Définition 4 (Vecteurs associés à la transformation par ondelettes de Haar)**
*D'après la remarque précédente, on pose les vecteurs associés à la transformation par ondelettes de Haar suivants :*

$$\mathcal{H}_a = \frac{1}{2}(1,1) \in \mathbb{R}^2 \text{ pour les espaces } (V_j)_{j\in\mathbb{Z}}$$
$$\mathcal{H}_d = \frac{1}{2}(1,-1) \in \mathbb{R}^2 \text{ pour les espaces } (W_j)_{j\in\mathbb{Z}}$$

**Définition 5 (Fonction filtre)**
*Soit $n \in \mathbb{N}$. Soient $b, x \in \mathbb{R}^n$. Soit $a \in \mathbb{R}^*$.*
*On définit un nouveau vecteur $y \in \mathbb{R}^n$, image de $x$ par la fonction $\Phi_{a,b}$ tel que :*

$$\forall i \in [1;n],\ y_k = \frac{1}{a}\sum_{i=1}^{k} b_i x_{k-i}$$

*On notera $\Phi$ la fonction filtre sur les espaces d'approximation et $\Psi$ celle sur les espaces de détails*

Avant d'étudier l'algorithme, il reste encore à définir une fonction permettant d'extraire un sous vecteur.

**Définition 6 (Fonction d'échantillonage)**
*On note $\downarrow_k$, pour $k \in [1;n]$ la fonction suivante :*

$$\downarrow_k : \mathbb{R}^n \longrightarrow \mathbb{R}^{\lfloor n/k \rfloor}$$
$$(x_1, x_2, ..., x_n) \longmapsto (x_k, x_{2k}, ..., x_{\lfloor n/k \rfloor k})$$

**Algorithme de Mallat** Considérons une image sous forme de matrice $M \in M_{n,m}(\mathbb{R})$, où l'on supposera que $n$ et $m$ sont deux multiples d'une puissance de 2, quitte à la normaliser en ajoutant des zéros.

Pour calculer $V_{-j}$ et $W_{-j}$, on applique la fonction filtre en premier lieu sur les lignes de $M$. On applique alors les fonctions $\Phi$ et $\Psi$ au vecteur $M_{i,[1;m/2^j]} = (M_{i,1}, M_{i,2}, ..., M_{i,m/2^j})$ puis on applique au résultat ainsi obtenu la fonction de sous-échantillonage $\downarrow_2$. En deuxième lieu, on applique aux colonnes notées $M_{[1;n/2^j],j}$ de cette nouvelle matrice ces mêmes fonctions.

L'algorithme, pouvant être considéré comme une fonction notée $TW$, pour calculer la transformée par ondelettes sur les espaces $V_{-j_V}$ et $W_{-j_V}$ est donc le suivant :

**Algorithme 1** Transformée par ondelettes d'une matrice

**Entrée(s)** une matrice $M \in M_{n,m}(\mathbb{R})$ et un entier $j_V \in \mathbb{N}$
- **si** $n$ ne divise pas $2^{j_V}$ **alors**
  - Ajouter $(n - n \,\%\, 2^{j_V})$ lignes de zéros à $M$
- **fin du si**
- **si** $m$ ne divise pas $2^{j_V}$ **alors**
  - Ajouter $(m - m \,\%\, 2^{j_V})$ colonnes de zéros à $M$
- **fin du si**
- **pour** $k = 0$ **à** $j_V - 1$ **faire**
  - **pour** $i = 1$ **à** $n/2^k$ **faire**
    - Affecter au vecteur $M_{i,[1;\frac{m}{2^{k+1}}]}$ le résultat de la fonction $(\downarrow_2 \circ \Phi)$ appliqué au vecteur $M_{i,[1;\frac{m}{2^k}]}$
    - Affecter au vecteur $M_{i,[\frac{m}{2^{k+1}}+1;\frac{m}{2^k}]}$ le résultat de la fonction $(\downarrow_2 \circ \Psi)$ appliqué au vecteur $M_{i,[1;\frac{m}{2^k}]}$
  - **fin du pour**
  - **pour** $j = 1$ **à** $m/2^k$ **faire**
    - Affecter au vecteur $M_{[1;\frac{n}{2^{k+1}}],j}$ le résultat de la fonction $(\downarrow_2 \circ \Phi)$ appliqué au vecteur $M_{[1;\frac{n}{2^k}],j}$
    - Affecter au vecteur $M_{[\frac{n}{2^{k+1}}+1;\frac{n}{2^k}],j}$ le résultat de la fonction $(\downarrow_2 \circ \Psi)$ appliqué au vecteur $M_{[1;\frac{n}{2^k}],j}$
  - **fin du pour**
- **fin du pour**

**Sortie(s)** La matrice $M$ ainsi obtenue

Avec l'image témoin, on obtient l'image suivante pour $V_{-1}$ i.e. $j_V = 2$ :

FIGURE 4 – Transformée par ondelettes de l'image témoin pour $j_V = 2$

D'après la définition 3, comme $\forall j \in \mathbb{Z}, V_{j-1} = V_j \overset{\perp}{\bigoplus} W_j$, on en déduit l'algorithme de reconstruction suivant :

**Algorithme 2** Reconstruction d'une matrice

**Entrée(s)** une matrice $M \in M_{n,m}(\mathbb{R})$ et un entier $j_V \in \mathbb{N}$
  **pour** $k = j_V - 1$ **à** $0$ **faire**
    **pour** $j = 1$ **à** $m/2^k$ **faire**
      Affecter au vecteur $M_{[1;\frac{n}{2^k}],j}$ la somme du résultat de la fonction $(\Gamma \circ \uparrow_2)$ appliquée au vecteur $M_{[1;\frac{n}{2^{k+1}}],j}$ et du résultat de la fonction $(\Omega \circ \uparrow_2)$ appliquée au vecteur $M_{[\frac{n}{2^{k+1}}+1;\frac{n}{2^k}],j}$
    **fin du pour**
    **pour** $i = 1$ **à** $n/2^k$ **faire**
      Affecter au vecteur $M_{i,[1;\frac{m}{2^k}]}$ la somme du résultat de la fonction $(\Gamma \circ \uparrow_2)$ appliquée au vecteur $M_{i,[1;\frac{m}{2^{k+1}}]}$ et du résultat de la fonction $(\Omega \circ \uparrow_2)$ appliquée au vecteur $M_{i,[\frac{m}{2^{k+1}}+1;\frac{m}{2^k}]}$
    **fin du pour**
  **fin du pour**
**Sortie(s)** La matrice $M$ ainsi obtenue

# IV. Un algorithme efficace ?

## 1) Étude de la complexité

**Proposition 4 (Complexité de la fonction $TW$)**
*Soit $j_V \in \mathbb{N}$. Pour une image de taille nxm pixels, la complexité de $TW_{j_V}$ est :*

$$\boxed{O\Big(nm + nm^2 + mn^2\Big)}$$

**Remarque 2**
*En utilisant les résultats des propriétés 8 et 9 on peut modifier la fonction $filter$ pour faire chuter cette complexité à $O(nm)$. Cependant cette modification serait valable uniquement pour les ondelettes de Haar.*

Voici quelques performances en temps de l'algorithme $TW_{j_V}$ pour des matrices carrées (l'image d'origine est la même, seule diffère la taille). Les calculs se sont fait avec la configuration suivante :
- CPU : Intel Code 2 CPU 2.40 GHz
- RAM : 3.6 Go

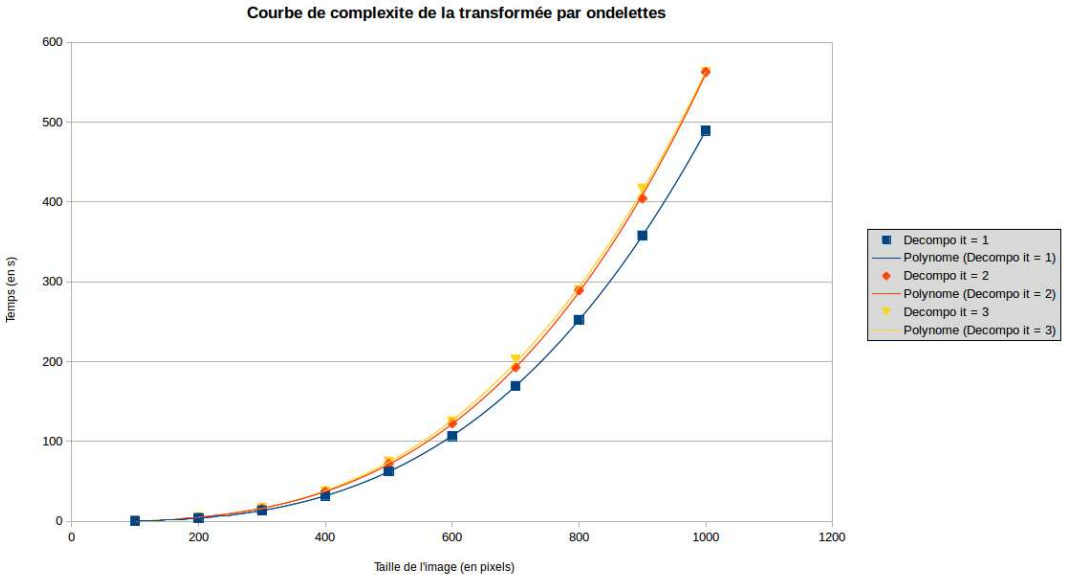


FIGURE 5 – Complexité de la fonction $TW_{j_V}$ en fonction de la taille de la matrice

## 2) Taux de compression

**Notation 1 (Nombre d'octets d'une image)**
*On notera dans la suite $No$ la fonction qui renvoie le nombre d'octets d'une image.*

**Définition 7 (Fonction de compression)**
*Ainsi on définit la fonction de compression pour $j_V \in \mathbb{N}$ comme mentionné ci-dessus :*

$$\gamma_{j_V} : M_{n,m}(\mathbb{R}) \longrightarrow M_{n/2^{j_V+1},m/2^{j_V+1}}(\mathbb{R})$$

*Où les entiers $n$ et $m$ sont supposés être des puissances de 2.*

**Définition 8 (Taux de compression)**
*Soit $j_V \in \mathbb{N}$. On définit alors la fonction taux de compression notée $\tau_C$ tel que :*

$$\begin{aligned} \tau_C : M_{n,m}(\mathbb{R}) &\longrightarrow [0;1] \\ e &\longmapsto \frac{No(\gamma_{j_V}(e))}{No(e)} \end{aligned}$$

Ainsi, on peut tracer le graphique suivant représentant le taux de compression $\tau_C$ où $it$ représente l'entier $j_V$. Les images utilisées sont carrées.

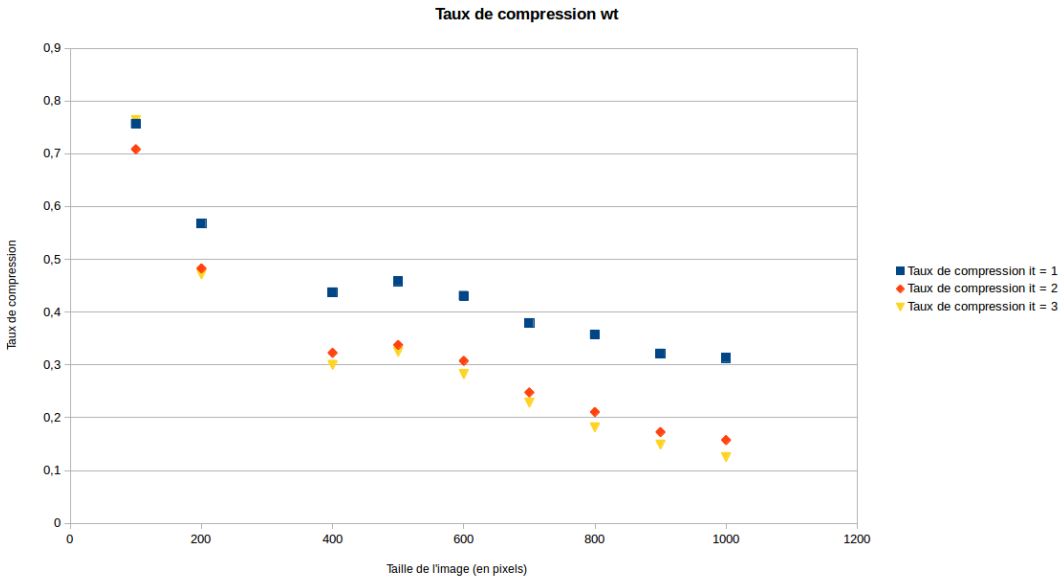


FIGURE 6 – Taux de compression $\tau_C$ en fonction de la taille de la matrice

**Remarque 3**

*Ainsi, d'après ce graphique, même si le temps requis est long, la compression semble plus efficace pour les grandes images que pour les petites. De plus, plus $j_V$ augmente, plus la compression est efficace (en effet, on divise à chaque fois la taille de l'image par 2).*

## 3) Estimation de l'erreur

Comme on l'a vu dans la partie précédente, la fonction $\gamma$ introduit une erreur qui provient du fait du passage d'une matrice de type $float$ au type $int$. De plus, l'appel à la fonction $Pixel\_matrix.intmatrix\_of\_matrix$ introduit une autre erreur qui est le passage d'un intervalle de $\mathbb{R}$ à l'ensemble $[0; 255]$.

On va donc introduire une fonction erreur. Pour cela, l'idée la plus simple est d'utiliser une fonction distance. Cette fonction distance devra respecter deux critères :

- prendre en compte la taille de la matrice
- prendre en compte tous les écarts relatifs entre les coefficients

**Proposition 5 (Distance de compression)**

*On définit par conséquent la distance suivante :*

$$\begin{aligned} d : M_{n,m}(\mathbb{R}) \, \mathrm{x} \, M_{n,m}(\mathbb{R}) &\longrightarrow \mathbb{R}^+ \\ (M, N) &\longmapsto \frac{\|M-N\|}{nm} \end{aligned}$$

**Définition 9 (Fonction erreur)**

*Soit $j_V \in \mathbb{N}$. On définit la fonction erreur comme suit avec $\gamma_{j_V}$ la fonction de compression et $\rho_{j_V}$ la fonction de décompression :*

$$\begin{aligned} \mathcal{E}_{j_V} : M_{n,m}(\mathbb{R}) &\longrightarrow \mathbb{R} \\ M &\longmapsto d\left(M, \left(\rho_{j_V} \, o \, \gamma_{j_V}\right)(M)\right) \end{aligned}$$

En reprenant l'image originale, on obtient alors :


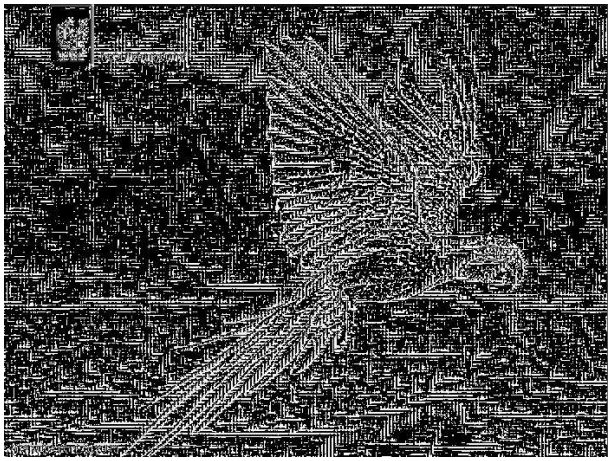

FIGURE 7 – Image décompressée issue de $\left(\rho_{j_V} \, o \, \gamma_{j_V}\right)$ (à gauche) et image représentant l'erreur commise (à droite)

# Conclusion

Ainsi, l'algorithme de Mallat permet de transformer une image par ondelettes. Lorsque cette transformation est faite, comme le poids de l'image ne change pas, il a été nécessaire de trouver une solution afin de pouvoir effectivement compresser une image. Cette solution produit deux fichiers, ainsi nécessaire lors de la transmission de ces données.

On a également vu que l'algorithme de Mallat n'introduit pas d'erreur, c'est uniquement dû à la conversion d'une matrice de type $float$ au type $int$ que l'erreur est introduite. De plus, cette erreur diminue lorsque la taille de l'image augmente. Tout comme le taux de compression. Cependant, la complexité temporelle est un sérieux frein, même si une complexité de l'ordre de $O(n^3)$ est acceptable pour un algorithme travaillant sur les matrices. En effet, la transformation requiert la plupart des coefficients de la matrice et la fonction $filter$ s'effectue en temps quadratique (que l'on pourrait diminuer en un temps linéaire dans le cas de l'ondelette de Haar). En titre de comparaison, la complexité du produit matriciel naïf (c'est à dire, l'algorithme issue de la définition) a une complexité de $O(n^3)$.

Enfin, lorsque l'on augmente le nombre d'itérations de l'algorithme de Mallat (i.e. le nombre $j_V$), le taux de compression augmente mais l'erreur également. Il est donc peut être profitable de rester dans des petites valeurs de $j_V$ afin de garder l'intégrité et la cohésion des images.

Par conséquent, l'algorithme proposé ici est plus efficace pour les grandes images plutôt que les petites, même si l'algorithme s'effectue dans un temps plus long. Il faut donc trouver un compromis pour chaque taille d'image. Cependant, les problématiques actuelles de compression concernent surtout les grandes images et non les petites.

# Bibliographie


[1] Pascal Szacherski, *Compression, ondelettes et algorithmes afférents*, Janvier 2008

[2] Phillip K. Poon, *Wavelets*, College of Optical Sciences, University of Arizona, 2012

[3] Philippe Carré et Rémi Cornillet, *Code Matlab*, respectivements professeur de l'université de Poitiers et responsable de l'équipe Icones, étudiant à l'École Normale Supérieure de Rennes

[4] René Alt, *La transformation en ondelettes*, Professeur à l'université Pierre et Marie Curie

[5] Olivier Rioul, *Ondelettes régulières : application á la compression d'images fixes*, Télécom ParisTech, 1993

[6] Marc Lorenzi, *Code pour les images bmp*, Professeur au lycée Camille Guérin

# Annexe A


FIGURE A.1 – Image originale

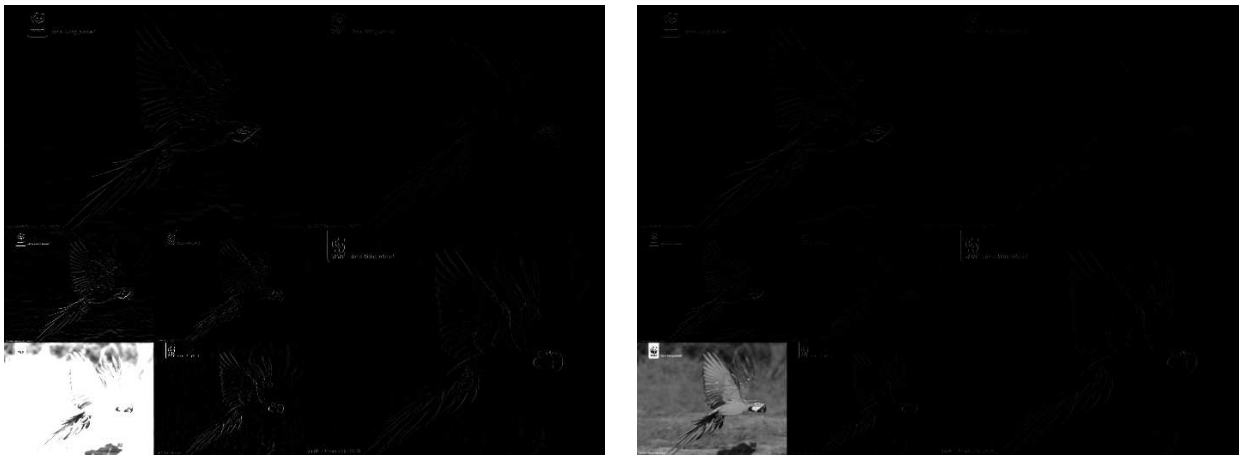

FIGURE A.2 – Exemple de la différence entre le choix du scalaire $\frac{1}{\sqrt{2}}$ (à gauche) et du scalaire $\frac{1}{2}$ (à droite)

On voit ainsi très clairement que le scalaire $\frac{1}{2}$ permet de garder une bonne luminosité de l'image originale.