R08921053 電機研一 梁峻瑋同學 TFW—HW5

(1) What are the vanish moments of (a) $\psi(t)$ where $FT[\psi(t)] = (f^3+f^4)\exp(-f^4)$.(b) the sinc wavelet and (c) the 8-point symlet, (d) the 18-point coiflet?
回答:
(a) 首先 $m_k=1/(-j2\pi)^k*d^k(FT[\psi(t)])/df^k|_{f=0}$, 對所有正整數 k 成立.
$m_0= FT[\psi(t)|_{f=0}=(f^3+f^4)\exp(-f^4)|_{f=0}=0$, 因為 $\exp(-f^4)=1$ when f=0.
$m_1=1/(-j2\pi)^1*d^1(FT[\psi(t)])/df^1|_{f=0}=C*(3f^2+4f^3+\text{higher term})*\exp(-f^4)|_{f=0}=0$.
$m_2=1/(-j2\pi)^2*d^2(FT[\psi(t)])/df^2|_{f=0}=C*(6f+12f^2+\text{higher term})*\exp(-f^4)|_{f=0}=0$.
$m_3= C*(6+24f+\text{higher term})*\exp(-f^4)|_{f=0}=1/(-j2\pi)^3*6*1 \mathrel{!=} 0$
所以 vanish moment 是 3.

(b) 由於 sinc wavelet 在-0.25~0.25 之間, 也就是(-0.25, 0.25), 值都是 0, 也因此和任意多項式相乘後還是 0, 積分後也只能是 0, 所以 **vanish moment 是無限大**.

(c) 從物理上的意義來看, vanish moment of Symlet = Daubechies wavelet, 但是 filter 更加的對稱, 所以 2p=8 點 symlet wavelet 有 **vanish moment = 4**.

(d) 從物理上的意義來看, 6p=18 點的 Coilfet wavelet 有 **vanish moment = 3**.

(2) Why the complexity of the 1-D discrete wavelet transform is linear with *N*?
回答:
在 1-D discrete wavlet transform 當中, 基本上就是要做兩次 convolution. 然而, x[n]和 g[n]的 convolution 是 Θ(N)的複雜度, 其中 N 是 x[n]的長度.

由於 g[n]通常是 12 點以內, 而且 convolution 基本上是相加的運算, 所以可以分段計算, 一次只計算 N_1點, 時間複雜度=$\Theta(N_1+12-1)\log(N_1+12-1))=\Theta(N_1\log(N_1))$, 累積 N/N_1 次以後, 就是 Θ(N)的複雜度, 因為 $\log(N_1)$是常數

(3) Why the wavelet transform can be used for (a) pattern recognition and (b) filter design, and (c) image compression? .
回答:
(a) 首先, 由於 wavelet transform 可以在保留特徵的前提下壓縮圖片，所以我

們可以先在壓縮後的小圖上，判斷大範圍的特徵．接下來再在壓縮後的中圖片上，判斷中範圍的特徵．最後再在原本的大圖上，判斷小範圍的特徵.

(b) 由於 wavelet transform 可以過濾掉高頻的訊息，所以也可以把它當作 low-pass filter 來看待，使用它來做 filter design，而且更能夠保留 edge 的特徵.

(c) Wavelet transform 可以用來做影像壓縮，因為做完 wavelet transform，低頻部分會保留原特徵，高頻部分只會留下變化大的部分，而且尺寸會變成 1/2*1/2，所以兩軸都是低頻的成品可以當作影像的壓縮．至於一軸低頻一軸高頻的部分，則是可以用來偵測邊緣，因為高頻的那軸上變化的部分會顯現出來.

(4) For a two-point wavelet filter, if *g*[0] = *a*, *g*[1] = *b*, and *g*[*n*] = 0 otherwise,
(a) What are the constraints of *a*, *b* if *g*[*n*] is a quadratic mirror filter?
(b) What are the constraints of *a*, *b* if *g*[*n*] is an orthonormal filter?
回答:
(a) 已知 g[n]，所以 $G(z)=a+bz^{-1}$, $G(z)^2=aa+2abz^{-1}+bbz^{-2}$，我們有:

$$\det\left(H_m(z)\right)=G(z)H(-z)-H(z)G(-z)=G(z)^2-G(-z)^2=4abz^{-1}=2z^k$$

所以, k=-1，而且**(a,b)必須要滿足"a*b=0.5"的條件限制**.

(b) $G(z)=a+bz^{-1}$, $G_1(z)=G(z^{-1})=a+bz$, $G(z)G_1(z)=aa+bb+abz+abz^{-1}$，我們有:

$$G_1(z)G_1(z^{-1})+G_1(-z)G_1(-z^{-1})=G(z)G_1(z)+G(-z)G_1(-z)=2(aa+bb)=2$$

所以, **(a,b)必須要滿足"$a^2+b^2=1$"的條件限制**.

(5) What are the main advantages of (a) the symlet and (b) the contourlet?
回答:
(a) Symlet wavelet transform 的優點在於, **g[n]的最大值幾乎在中間，所以做 wavelet transform 以後，圖片偏移的幅度很小**，也因此被稱作 symmetric wavelet.

(b) Contourlet wavelet transform 的優點在於，**由於它不是沿著 x 軸/y 軸做小波轉換，所以各個方向的直線邊緣變化(高頻成分)都可以偵測到**.

(6) (a) Write a Matlab program for the following 2-D discrete 10-point Daubechies wavelet.
回答:
程式碼如下

```
function y = wavedbc10(x);
%Step1: to set g[n] and h[n]
g = [0.0033 -0.0126 -0.0062 0.0776 -0.0322 -0.2423 0.1384 0.7243 0.6038 0.1
gT = transpose(g);
h = [0.1601 -0.6038 0.7243 -0.1384 -0.2423 0.0322 0.0776 0.0062 -0.0126 -0.
hT = transpose(h);
%Step2: to compute v1L[m,n] and v1H[m,n]
xg = conv2(g, x);
xgT = transpose(xg);
v1L = transpose(downsample(xgT, 2));
xh=conv2(h, x);
xhT = transpose(xh);
v1H = transpose(downsample(xhT, 2));

%Step 3: to compute x1L[m,n] and x1H1[m,n]
v1Lg = conv2(gT, v1L);
x1L = downsample(v1Lg, 2);
v1Lh = conv2(hT, v1L);
x1H1 = downsample(v1Lh, 2);

%Step 4: to compute x1H2[m,n] and x1H3[m,n]
v1Hg = conv2(gT, v1H);
x1H2 = downsample(v1Hg, 2);
v1Hh = conv2(hT, v1H);
x1H3 = downsample(v1Hh, 2);
y = [x1L, x1H1, x1H2, x1H3];
```

解釋:

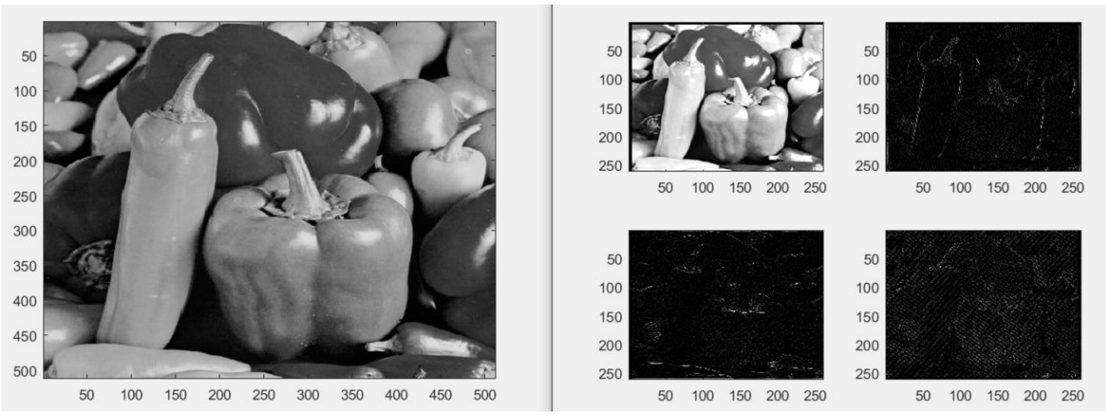
Step1: 設定 Daubechies wavelet 的 g[n]和 h[n]. 由於下一步的 downsample 指令是針對列向量做的, 所以可能需要轉置, 要計算 gT 和 hT.

Step 2: 計算 v1L 和 v1H, 原則上就是用 conv2()指令, 但是 downsample()是對列向量做的運算, 所以把 downsample 的輸入和輸出都做轉置的處理.

Step 3: 計算 x1L 和 x1H1, 就是用 conv2, 再搭配 downsample().

Step 4: 計算 x1H2 和 x1H3, 同 Step 3.

以下是輸出的圖片:



(b) Also write the program for the inverse 2-D discrete 10-point Daubechies wavelet transform.

回答：程式碼如下

```
function y = iwavedbc10(x1L, x1H1, x1H2, x1H3)
g1 = [0.1601 -0.6038 0.7243 -0.1384 -0.2423 0.0322 0.0776 0.0062 -0.0126 -0.0033];
h1 = [-0.0033 -0.0126 0.0062 0.0776 0.0322 -0.2423 -0.1384 0.7243 -0.6038 0.1601];
h1T = transpose(h1); g1T = transpose(g1);

X1L = upsample(x1L, 2);
X1H1 = upsample(x1H1, 2);
x0 = conv2(X1L, g1T)+conv2(X1H1, h1T);

X1H2 = upsample(x1H2, 2);
X1H3 = upsample(x1H3, 2);
x1 = conv2(X1H2, g1T)+conv2(X1H3, h1T);

x0T = transpose(x0);
X0 = transpose(upsample(x0T, 2));
x1T = transpose(x1);
X1 = transpose(upsample(x1T, 2));
newx = conv2(X0, g1)+conv2(X1, h1);
for i=1:10
    newx(1, :)=[]; newx(:, 1)=[];
end
L = size(newx, 1);
for i=1:9
    newx(L-8, :)=[]; newx(:, L-8)=[];
end
y = newx;
```

Step 1:設定 h1, h1T:=h1^T, g1, g1:=g1^T

Step 2:先 upsample，再和 g1T/h1T 做 conv2 還原，最後相加．基本上就是反運算.

Step 3:同上，這次對 x1H2, x1H3 做.

Step 4:同上，這次對 x0, x1 做；由於 upsample 只對列向量運算，所以輸入和輸出都必須要先做 transpose 來處理，最後再 conv2 後相加就得到 newx.

Step 5:有 newx 以後，要把四周邊緣為 0 的列和行給移除，最後再輸出.

輸出的圖片如下: