**Homework #3, due June 30**

**FFT packages:** http://www.netlib.org/fftpack, vfftpack, fishpack;
**Matlab Demo:** Toolboxes, Image Processing, Wavelet Toolbox (wvdtool)
**Further Reading on Wavelets:** Prof. L-Liang Chern: An Introduction to Wavelets, available at http://www.math.ntu.edu.tw/˜chern. This is a quite readable lecture note, recommended for further study on mathematical issues such as regularity and convergence rate, etc.
**Alternative Wavelet Package (.m files):** By Prof. Chien-Chang Yen. Available upon email request (wangwc@math.nthu.edu.tw).

Paper and Pen Assignments

1. Verify equation (41) of the lecture note.
2. Verify that (47) implies (41).
3. Show that if $a_l$ has compact support and the Cascade Algorithm converges, then (54) holds for $\phi^n$ for all $n$ large enough and hence for the limit $\phi$.
4. Verify that the Haar wavelet functions $\{\phi_{0,k}, \psi_{0,k}, \cdots, \psi_{J-1,k}\}$ constitute a basis for $l^2_{\{\frac{1}{2^J}, \frac{2}{2^J}, \cdots, 1\}}$.
5. Give an estimate of operation count (multiplications) for wavelet decomposition (obtaining the coefficients $c_{\cdot,\cdot}$ and $d_{\cdot,\cdot}$ from $f(x)$, $x = \frac{1}{2^J}, \frac{2}{2^J}, \cdots, 1$) and wavelet reconstruction (vice versa). For simplicity, you may take the Haar wavelet. At the finest level, $d_{J-1,k} = \langle f, \psi_{J-1,k}\rangle$ can be computed efficiently using direct summation. (what about other wavelets?)

Programming Assignments

6. Explore the ability of wavelets of being able to reveal features of a signal not only in frequency or time domain but on both domains together.

   Consider the periodic signal:

$$f(t) = \begin{cases} \cos(t) + C & \text{for } t \in [-1, 0] \\ \sin(t^6)/t^3 & \text{for } t \in [0, 2] \end{cases}$$

   where $C = \sin(2^6)/2^3 - \cos(1)$, sampled at 1024 equidistant points in [-1 2].

   **(a)** Use the discrete cosine transform to compress the signal.

   Use about 10% of the leading discrete cosine coefficients (low pass filter). Plot the "compressed signals" $\tilde{f}_j$ together with the original signal. Find the "relative energy error" $\sum |f_j - \tilde{f}_j|^2 / \sum |f_j|^2$. Describe the behavior of the approximation in the vicinity of the discontinuity at $t = 0$ (known as the Gibbs phenomenon), in the high frequency part ($t > 1.5$), as well as in the smoothly varying region.

   Repeat the above compression process by using 10% of the discrete cosine coefficients of the largest absolute values instead of the leading discrete cosine coefficients.

**(b)** Use the wavelet transform to compress this signal.

Use about 10%, of the wavelet coefficients (use the coefficients with the largest absolute values) for different wavelet models. Try "db1" (Haar) first. If you have access of other source codes, you can also try "db2" (Daubechies, 4 coeff.) and "db4" (Daubechies, 8 coeff.).

Plot the "compressed signals" together with the original signal. Find the "relative energy errors" and describe the behavior of the approximation. You may also get some information in matlab by using command `waveinfo('db')`)

If you have access to matlab with 'Image Process' toolbox (not on our course machine, unfortunately), you may write your own discrete wavelet transformation code. The filter bank data can be found in matlab by using command `[LO_D,HI_D,LO_R,HI_R]= wfilters('db2')`. Some useful matlab commands: `dwt` for Single-level discrete 1-D wavelet transform; `idwt` for Single-level inverse discrete 1-D wavelet transform; `wavedec` for Multi-level 1-D wavelet decomposition; `waverec` for Multi-level 1-D wavelet reconstruction. Some other useful matlab commands: `help wavelet, wmaxlev, sort, abs`, Wavelet Toolbox (wvdtool), Matlab demo (wavedemo).

Otherwise, you can use the wavelet package by Prof. Chien-Chang Yen.

**(c)** Consider a signal $f(t) = \sin(5t)$. Obtain the FFT and the discrete wavelet transform (DWT) (using 'db?' for example) of this signal. Plot the signal as well as its FFT and DWT. Next, change the signal $f(t)$ by making a very small section of it equal to zero. Get again the same three plots and compare them with the first ones. Comment on the differences. What information about the signal alteration can we get from the FFT and the DWT?.

7. Explore image compression and Wiener Filter

An "image" is a function defined on $(a, b) \times (c, d)$ which gives the brightness at each point. A discretized image consists of "pixels" obtained by dividing $(a, b) \times (c, d)$ in $N$ by $M$ subsquares, with a brightness value given in the center of each subsquare. The two dimensional version of the discrete Cosine transform (`dct2` and `idct2` in Matlab) is frequently used in image processing.

**Remark**: We only have one dimensional discrete cosine and inverse cosine transforms `dct` and `idct` on the course machine. The `dct2` and `idct2` can be generated using `dct` and `idct` and matrix transposition. (How?)

**(a)** (import an image into matlab). Take an image (it should be sharp and have strong contrast) in standard image format (GIF, BMP, TIFF, etc). However, you should not use JPEG format since JPEG images have been modified in a way such that almost surely the higher frequency DCT coefficients were zeroed out. You can for example download an image from web, or scan in an image and then save it either in GIF or TIFF format (TIFF can store more image information for each pixel).

Select a part of your image which is of size 256 by 256. You can use `xv` in Unix system to cut the image, or select a smaller rectangle later from Matlab (say `im=im(1:256, 1:256)`).

(Type `xv &`. Click the right mouse button to bring up the controls and load your image. Select "Image Info" from the "Windows" menu. While holding the left mouse button drag the mouse from one corner to the diagonally opposite corner while the current size is displayed in the Info window. When you are satisfied with the selected area release the mouse button, then click "Crop". Click the Save button, choose as format "TIFF" and as colors either in "Greyscale" (one array) or full color (RGB, Red, Green and Blue, 3 arrays). Select "Compression: None")

Import the image into Matlab as a two dimensional array `im` using the command

```
im = double(imread('filename.tif','tif'))
```

and display your image using the command

```
imshow(im,[])
```

**Remark**: Unfortunately, we don't have `imshow` on the course machine. You can use `imwrite` together with `xv` to view the image.

**(b)** (image compression)

The image is denoted as $u_{i,j}$, the discrete cosine transform of this image is denoted as $\hat{u}_{k,\ell}$. The quality of the image can be reasonably retained even if only few terms of discrete cosine coefficients in $\hat{u}_{k,\ell}$ are used to reconstruct the image. This is known as image compression. The rate of the compression is the percentage, $x\%$, of the discrete cosine coefficients used to reconstruct the image. In other words, we can set $(1 - x\%)$ of the less significant amplitude discrete cosine coefficients to zero. (you may use matlab commands: `dcim=dct2(im);`, (suppose the size of dcim is $N \times N$), use `dcim1=reshape(dcim, N*N,1)` to get one dimensional array; `[s, index]=sort(abs(dcim1)); cutoff=(1-x%)*N*N, dcim1(index(1:cutoff))=0`, dcim= reshape(dcim1, N, N)). Find out the largest possible compression such that the compressed image still resembles original image well. (you can also use the Matlab demo, Toolboxes, image processing, discrete cosine transformation to see the effects in the data compression)

**(c)** (recover a "blurred" image)

A blurred image, denoted as $u_b$, can be constructed by replacing the value of each pixel with the average value of all pixels in a 15 by 15 rectangle. This kind of filter can be defined using the matrix:

$$f = \frac{1}{225}\begin{bmatrix} I_{15,15} & 0 \\ 0 & 0 \end{bmatrix}_{\text{size of } u}, \quad I_{15,15} = \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}_{15,15}$$

(In matlab, $I_{15,15}$= `ones(15)`). The "blurred" image is then constructed by convoluting the original image $u$ by the filter $f$.

It's easiest to do it in the discrete cosine domain because convolution becomes simple multiplication:

$$\hat{u}_b = \hat{u}\hat{f}$$

(in matlab, use command: $\hat{u}_b$ =$\hat{u}$.*$\hat{f}$) and $u_b$ is the inverse transformation from $\hat{u}_b$.

Test that you can recover the original image by deconvolution:

$$\hat{u} = \hat{u}_b/\hat{f}$$

(in matlab use command: $\hat{u} = \hat{u}_b$ ./ $\hat{f}$) and $u$ is the inverse transformation of $\hat{u}$.

**(d)** (Wiener filter for noise) We can now add a random noise (i.i.d.) at each pixel of the blurred image $u_b$. The distribution of the noise can be uniform distribution Unif[-c,c] where $c$ is equal to 4% of the maximal pixel value in the blurred picture, or normal distribution with mean zero and standard deviation $\sqrt{c/6}$.

The original picture can be recovered by deconvolution and Wiener filter (also known as optimal filter) as follows:

$$\hat{v} = \frac{\hat{u}_n\hat{f}}{\hat{f}^2 + \alpha^2}$$

(in matlab use command: $\hat{v} = (\hat{u}_n \;.\!*\; \hat{f}) \;./\; (\hat{f} \;.\!*\; \hat{f} + \alpha^2))$ where $\hat{u}_n$ is the transformed blurred and noisy image, $\hat{f}$ is the transformed filter defined in the previous question, $\alpha$ is a free parameter that should be optimized for each case.

Try to recover the original image and report the best possible parameter $\alpha$ you have used. (you can also use `wiener2` in matlab, the Matlab demo, Toolboxes, image processing, noise reduction filtering to see the effects of the noise and filtering)

**(e)** (Optional, harder.) In part (b) and (d), the human eyes (more precisely, your eyes) is the final judge whether a compressed or denoised image is satisfactory. We may not be able to afford the time for parameter tuning and human inspection in real applications. In this subproblem, I would like to challenge you with all possibilities to resolve the issues. We have a blurred image due to hand shaking. If the blurring were due to constant motion of the object or the camera (high speed vehicle or automatically rotating camera), we may be able to get some (or near all) information for the shape of the convolution kernel. It is not clear how much partial information we have in the case of hand shaking. If we only have partial information, we may need

(i) Human eyes inspection or quantitative measurement of the 'clarity (blurring)' in order to determine whether the deblurring is satisfactory.

(ii) An optimization process to maximize the clarity/satisfaction.

The image is on the course homepage and you are welcome to explore your ideas and discuss with your classmates or me anytime. I don't have a definite answer to this problem yet, so enjoy the problem.