Cognitive Science 276 – Neuroimaging

Homework #2: Fourier Transform and Image Reconstruction

The goal of this homework is to learn learn basic concepts of 1-D and 2-D Fourier analyses in MATLAB, including forward and inverse Fourier transforms (FT) of signals in the spatial and frequency domains, amplitude and phase spectrums, sampling, image reconstruction, aliasing of images, and ghosts. *(Hint: matrix entries are conventionally identified by (row,column), which is up/down, then left/right, while pixels are conventionally identified by (x,y), which is left/right, then up/down)*.

1: 1-D FT. Plot the following 1-D functions and their amplitude spectra:

$f_1(x) = 1.0*cos(3*2\pi*x)$, $f_2(x) = 0.25*cos(25*2\pi*x)$, and $f_3(x) = f_1(x) + f_2(x)$

Qualitatively describe the spectrum of the first two functions and then the spectrum of their sum. *Hint:*

```
step=1/256;        % [stuff after percent is a comment]
x=0:step:1-step;   % get 256 values of x from 0 to 1
f1=cos(9*2*pi*x);  % f1 is a real vector with 9 cycles from 0-1
FT_f1=fft(f1);     % fft() returns the fourier transform
plot(abs(FT_f1));  % FT_f1 is complex vector; abs() gets element amplitudes
```

2: 1-D Inverse FT. (this is a continuation of Problem 1). $F_1(f)$ is the spectrum of $f_1(x)$. Reconstruct a function $f'_1(x)$ from $F_1(f)$ using the inverse Fourier transform, `ifft()`. Plot the real and imaginary parts, and the amplitude and phase of $f'_1(x)$. using `real()`, `imag()`, `abs()`, and `angle()`. Explain why the imaginary and the phase plots look 'funny'.

3: Display 2-D Image. Convert a 2-D tiff image into a matrix with: `B=double(imread('t1sag.tiff'));` Download 256x256 tiff (sagittal T1 brain image) from `http://cogsci.ucsd.edu/~sereno/276/t1sag.tiff` (other uncompressed, grayscale tiffs work, too). Plot it with: `colormap gray; imagesc(B,[minB maxB]); axis square;` Play with minB and maxB (don't forget the brackets). It autoscales if you omit the `[minB maxB]`.

4: 2-D FT. Compute the 2-D Fourier transform of image *B* using `F=fftshift(fft2(B));` Then make four plots of *F* (which is a 2-D matrix of complex numbers): first the real and imaginary components, and then the corresponding amplitude and phase components. Use the functions: `real(), imag(), abs(), angle()` and `imagesc(Comp,[minComp maxComp]);` You will have to experiment with the minimum and maximum values to make sense of the pictures. You can also use the functions `min()` and `max()` (apply them twice to operate on a matrix). Describe the resulting distribution of energy in spatial frequency space (k-space).

Hint (image and its amplitude spectrum plotted on one page):

```
figure;        % multiple plots on one figure
B = double(imread('t1sag.tiff'));
F = fftshift(fft2(B));
A = abs(F); minA = min(min(A)); maxA = max(max(A));
colormap gray;
subplot(1,2,1); imagesc(B);                  axis square; title('Real Image');
subplot(1,2,2); imagesc(A,[minA 0.1*maxA]); axis square; title('K-Space Amplitude');
```

5: K-space Center. Manipulate the amplitude of the center point of k-space: $F(k_{x_0},k_{y_0})$, which in the present case can be referenced with `F(129,129)` (N.B.: the coordinates given for the center of K-space assume `fftshift()` has *first* been applied). First *double* the center point, reconstruct the images by using `ifft2(ifftshift())`, plot the original and reconstructed *amplitude* image using the same maximums and minimums, and describe the result. Then do the same reconstruction after *zeroing* the center point.

6: Spikes in K-space. An individual data point in K-space is sometimes mistakenly assigned a very large value (e.g., as a result of an electrical transient at the exact moment that the data point was being collected). Modify the following three K-space points by setting them to a large value (e.g., $5*10^6$), one at a time:

(a) F(129+4, 129) (k_y=4, k_x=0)
(b) F(129, 129+20) (k_y=0, k_x=20)
(c) F(129+4, 129+20) (k_y=4, k_x=20)

Reconstruct the images in each case using `ifft2(ifftshift())` and plot and describe them.

7: Zero Portions of K-space. By setting portions of K-space to zero, certain ranges of spatial frequency will be removed when the image is reconstructed. Set the following regions of K-space to zero, reconstruct the images as above, then plot K-space (amplitude) and the reconstructed image in each case, and comment on the result:

(a) set F(k(y),k(x)) = 0, where both x and y are between 129-32 and 129+32 (high pass)
(b) set F(k(y),k(x)) = 0, *except* where x and y are between 129-32 and 129+32 (low pass)
(c) set F(k(y),k(x)) = 0, when x is between 130 and 256 (zero right half k-space)

Hint: make blank mask using `ones()` *or* `zeros()`, *set ranges of mask to 0 or 1 using* `low:high` *syntax, apply mask to K-space with element-wise operators (e.g., for multiplication:* `.*` *).*

8: Subsample K-space. If an *image* (or a time signal) is not sampled frequently enough, aliasing (wraparound) will occur in the *frequency* domain (that is, after a Fourier transform). This is also true when going from the *frequency* domain back to *space* (or time); that is, if *K-space* is not sampled frequently enough, aliasing will result in the *image* (or time) domain. Simulate this by zeroing K-space points:

(a) set every K-space point whose x coordinate has an *even* number to zero (zero even numbered K-space lines)
(b) set every K-space point whose y coordinate has an *even* number to zero (zero lines in other direction)
(c) apply the operation in (a) to the K-space result you got in (b) (undersample *both* directions)

Comment on the effect of these three different undersamplings after reconstructing the images with `ifft2(fftshift())`.

9: Shift Alternate Lines of K-space. When K-space data is collected during an EPI scan, the even and odd lines may not be properly aligned because of imperfections of the gradients. Simulate this by shifting even K-space lines to the left and the odd K-space lines to the right:

(a) set F(k(x),k(y)) = F(k(x-1),k(y)), when y is odd and F(k(x+1),k(y)), when y is even
(b) set F(k(x),k(y)) = F(k(x-3),k(y)), when y is odd and F(k(x+3),k(y)), when y is even

Plot both K-space and reconstructed images for the above manipulations. How do the wraparound ghosts subtly differ from the ones generated in the previous problem?
Hint: watch limits so you don't go off the edge