

HOMEWORK #3
Due at the start of Class on Thursday 10/19/06

Readings:

Sections 2.2.3; 2.2.5, 2.2.6; 2.4, 2.5, 2.6, 3.2; 3.3; 6.3
Preview Section 2.8

Problems:

1. Use the convolution and modulation theorems to find and graph the transforms of the following functions: $\text{sinc}(x) * \text{sinc}(3x)$ and $[\text{sinc}(2x)\cos(20\pi x)]^2$
2. Let $g(x, y) = \exp(-j2\pi(5x + 2y))\cos(24\pi x)$. Find and sketch its 2D Fourier Transform. Hint: the function is separable.
3. Define the object $m(x, y) = \text{rect}(x/2)\text{rect}(y/2)$
 - a) Sketch the sinogram for this object. Give explicit expressions for the projections at 0, 90 degrees, and 45 degrees (e.g. the projection that goes through the diagonals of the square). For other angles, either a sketch or an explicit expression is okay.
 - b) Show that the projection theorem holds for $\theta = 45$ degrees. In other words, derive the 2D Fourier transform and the projection at $\theta = 45$ degrees and show how these are related.
4. Problem 6.8
5. Problem 6.12

Matlab Exercise: The purpose of this exercise is to familiarize you with the MATLAB functions used for performing 2D Fourier transforms and manipulating and displaying images. **Boldfaced** items should be turned in with the homework. You can always get more information about a command by typing *help <name of command>*, e.g., *help fft2*.

Steps:

1. First download the file BE280Ahw1im.mat from the course website.
2. Load the image into MATLAB with the command: *load BENG280Ahw1im*.
3. Type *whos* to see the variables in your MATLAB workspace. You should see a variable named *Mimage*. Type *size(Mimage)* to see how large the image is.
4. Use the command *imagesc(Mimage)* to display the image – you should see a sagittal image of a head. To change the colormap display to gray-scale, type *colormap(gray(256))* which will result in a display with 256 shades of gray. **Print out a copy of the image.** Try experimenting with different numbers for the colormap, e.g. *colormap(gray(20))*;
5. Compute the 2D Fourier transform of the image with the command $Mf = \text{fft2}(Mimage)$; where the 2D transform will now be stored in the variable *Mf*. Remember to add the semi-colon at the end of the command, otherwise MATLAB will display all the numbers in the matrix! The command *fft2* puts the zero-frequency value of the transform at the first indices

of the matrix. For display it's convenient to put the zero-frequency value in the center of the matrix. To do this, type $Mf = \text{fftshift}(Mf)$;

6. Type $\text{imagesc}(\text{abs}(Mf))$ to display the magnitude of the transform. It will be hard to see anything, because the dynamic range of the Fourier transform is so large. To get a sense of the dynamic range, find the minimum value of the Fourier Transform with the command $\text{min}(\text{min}(\text{abs}(Mf)))$ and the maximum value with the command $\text{max}(\text{max}(\text{abs}(Mf)))$. **Record these minimum and maximum values. What is the ratio of the maximum to minimum value?**
7. To scale the image so that you can get a better sense of what the Fourier transform looks like, you can use the imagesc command with the syntax: $\text{imagesc}(\text{abs}(Mf), [cmin\ cmax])$ where $cmin$ will be displayed as the darkest value on the image and $cmax$ will be displayed as the lightest value on the image. For example, typing in $\text{imagesc}(\text{abs}(Mf), [200\ 1e6])$ should give you a nice looking result. Experiment with different values of $cmin$ and $cmax$.
8. You can also look at the real part, the imaginary part and the phase of the transform with the commands $\text{imagesc}(\text{real}(Mf))$, $\text{imagesc}(\text{imag}(Mf))$, and $\text{imagesc}(\text{angle}(Mf))$, respectively. If necessary you can use the $[cmin\ cmax]$ option to scale the image properly. **Print out images of the magnitude and phase and real and imaginary parts of the transform. To plot more than one image on the same Figure, you can make use of the subplot command in MATLAB.**
9. Another way to look at the transform is to use the mesh command. Try $\text{mesh}(\text{abs}(Mf))$. It will be a little hard to see what is going on, so do the following: Define $\text{span} = 128 + (-20:20)$; then type $\text{mesh}(\text{abs}(Mf(\text{span}, \text{span})))$;
10. **Resolution.** What happens when we zero out the outer regions of the Fourier transform?
 - (a) *Resolution reduction in the x-direction.*

```
>> res_span = 129+(-16:16);  
>> Mf2 = zeros(256,256);  
>> Mf2(:,res_span) = Mf(:,res_span);  
>> Mf2 = fftshift(Mf2);  
>> M_resx = ifft2(Mf2);  
>> imagesc(abs(M_resx)); % This will show reduction of resolution in the x-direction.
```
 - (b) Demonstrate resolution reduction in the y-direction. **Hand in code and image.**
 - (c) Demonstrate resolution reduction in the x and y directions. **Hand in code and image**
11. **Missing data in k-space.** We can also zero out the inner regions of the Fourier Transform.

```
>> Mfzero = Mf;  
>> Mfzero(ky,kx) = 0;  
>> iMFzero= ifft2(fftshift(Mfzero)); % look at resulting image.
```

Zero out the following:
 - (a) $kx = 129$; $ky = 129$
 - (b) $kx = 1:256$; $ky = 129+(-16:16)$;
 - (c) $kx = 129+(-16:16)$; $ky = 1:256$;
 - (d) $kx = 129+(-16:16)$; $ky = 129 + (-16:16)$;
 - (e) $kx = 1:2:256$; $ky = 1:256$;**For each set of parameters, plot out the Fourier transform and the resulting image. Give a qualitative explanation of why the image looks the way it does.**
12. **Spikes in the data.** You can put a spike at location (kx,ky) in Fourier space with the following commands

```
>>Mfspike = Mf;  
>>spike = 100e6;  
>>Mfspike(ky,kx) = Mf(ky,kx) + spike; % add spike  
>>iMFspike = ifft2(fftshift(Mfspike)); % look at resulting image.
```

Put spikes at:

- (a) $k_x = 129$; $k_y = 129$ – note this point corresponds to the center of k-space (i.e. $k_x = 0$, $k_y = 0$ in terms of the coordinates we used in class).
- (b) $k_x = 161$; $k_y = 129$
- (c) $k_x = 161$; $k_y = 161$

For each spike location, plot out the Fourier transform and the resulting image. Give a qualitative explanation for why the image looks the way it does. For example, what accounts for the direction of the artifacts?