HW2 - send Matlab file by midnight October 15.


**Fourier transforms**

The 1-dimensional fourier transform is defined as:

\[
F(k) = \int_{-\infty}^{\infty} f(x)e^{-i2\pi kx} dx \quad F(k) = \mathcal{F}[f(x)] \quad \text{forward transform}
\]

\[
f(x) = \int_{-\infty}^{\infty} F(k)e^{i2\pi kx} dk \quad f(x) = \mathcal{F}^{-1}[F(k)] \quad \text{inverse transform}
\]

where \( x \) is distance and \( k \) is wavenumber where \( k = 1/\lambda \) and \( \lambda \) is wavelength.

**Fourier series (including dimensions)**

There are \( N \) equally-spaced points over the length interval \( L \) so the spacing is \( dx = L/N \). The wavenumbers are \( k_n = m/L \).

\[
F_m = \sum_{n=0}^{N-1} f(x_n) \exp\left(-i2\pi \frac{m}{L} x_n\right) dx
\]

\[
f(x_n) = \sum_{m=-N/2}^{N/2-1} F_m \exp\left(i2\pi \frac{m}{L} x_n\right) dk
\]

We can simplify this a bit by noting that \( x_n = ndx = nL/N \) so the formulas become.

\[
F_m = \sum_{n=0}^{N-1} f(x_n) \exp\left(-i2\pi \frac{mn}{N}\right) dx
\]

\[
f_n = \sum_{m=-N/2}^{N/2-1} F_m \exp\left(i2\pi \frac{mn}{N}\right) dk
\]
These summations are the form of the \texttt{fft()} and \texttt{ifft()} in Matlab although there may be confusion regarding the normalization \(1/N\).

**Fourier transform of a Gaussian function**

\[
e^{-\pi k^2} = \int_{-\infty}^{\infty} e^{-\pi x^2} e^{-i2\pi ks} \, dx = \mathcal{F}\left[e^{-x^2}\right]
\]

**Some properties of fourier transforms**

- **Similarity property**
  \[
  \mathcal{F}\left[f(ax)\right] = \frac{1}{|a|} F\left(\frac{k}{a}\right)
  \]

- **Shift property**
  \[
  \mathcal{F}\left[f(x - a)\right] = e^{-i2\pi ka} F(k)
  \]

- **Differentiation property**
  \[
  \mathcal{F}\left[\frac{df}{dx}\right] = i2\pi k F(k)
  \]

**HW questions**

1. Write a program to generate a cosine function using \(2048\) points. Generate exactly \(32\), or \(64\) cycles of the function. Plot the results and add labels.

```matlab
figure(1)
clear
nx=2048;
kc=64/nx;
x=0:nx-1;
generate the function
y=cos(2*pi*x*kc);
figure(1)
plot(x,y);
xlabel('x');
ylabel('cos(x)');
pause
```

2. Take the fourier transform of the function that you made in problem 1. Use \texttt{fftsift} to shift the zero frequency to the center of the spectrum. Generate wavenumbers for the horizontal axis. Take the inverse FFT. Do you get what you started with? (don't forget to undo the \texttt{fftsift}.)
% figure(2)
 subplot(5,1,1),plot(x,y);
 xlabel('x')
 ylabel('cos(x)')

% generate the wavenumbers
 k=-nx/2:nx/2-1;
 cy=fftshift(fft(y));
 subplot(5,1,2),plot(k,real(cy));
 xlabel('k')
 subplot(5,1,3),plot(k,imag(cy));

% do the inverse FFT
 yo=ifft(fftshift(cy));
 subplot(5,1,4),plot(x,real(yo));
 xlabel('x')
 ylabel('cos(x)')
 subplot(5,1,5),plot(x,real(y-yo));
 xlabel('x')
 ylabel('difference')
 pause

% 3) Do problem 2 over using a sine function instead of a cosine function.

% 4) Show that the Fourier transform of a Gaussian function is a Gaussian function.
% Plot the difference between the fft result and the exact function.
% When you do this problem, it is best to make the Gaussian function an even function
% of x just prior to computing the fft(). If you do this then the transformed
% Gaussian will be real and even. Also you will need to scale the transform by
% the point spacing dx = L/nx.

clear
 figure(3)
 nx=2048;
 L=20;
 dx=L/nx;
 a=1.1;
 x=a*(-nx/2:nx/2-1)*dx;
 g=exp(-pi*x.*x);
 subplot(4,1,1),plot(x,g);
 axis([-4,4,-.5,1.1])
 xlabel('x')
 ylabel('Gaussian')

% generate the wavenumbers
 k=(-nx/2:nx/2-1)/L;
 cg=fftshift(fft(fftshift(g)))*dx;

% 5) Use this Gaussian example to demonstrate the stretch property of Fourier transform. The
% results should be compared in the wavenumber domain.

% 6) Use this Gaussian function to illustrate the shift property of the Fourier transform.
% The results should be displayed as a shifted Gaussian in the space domain.

% 7) Use the Gaussian function to demonstrate the derivative property of the Fourier
% transform. The analytic derivative of the Gaussian should be compared with the
% Fourier derivative in the space domain.