1. (Tomography\(^1\)) In computed tomography, our goal is to process projection data to recover an attenuation image. The attenuation image is described by the two-dimensional signal \(s(x,y)\), and the projection data are modeled as:

\[
p_\theta(t) = \int_{-\infty}^{\infty} s(t \cos \theta - r \sin \theta, t \sin \theta + r \cos \theta) dr,
\]

where the angle \(\theta\) ranges from 0 to \(\pi\) radians. The way in which these data are processed to form an image of \(s(x,y)\) is described in Figure 10.4 of the text and summarized below:

- For each projection angle \(\theta\):
  - Compute the Fourier transform of the projection data:
    \[
P_\theta(f) = \int_{-\infty}^{\infty} p_\theta(t)e^{-j2\pi ft} dt.
    \]
  - Filter by the frequency response \(H(f) = |f|:\)
    \[
    G_\theta(f) = |f|P_\theta(f).
    \]
  - Compute the inverse Fourier transform to form the filtered projection:
    \[
g_\theta(t) = \int_{-\infty}^{\infty} G_\theta(f)e^{j2\pi ft} df.
    \]
- Form an attenuation image by summing the filtered projections over all angles:
  \[
s(x,y) = \int_{0}^{\pi} g_\theta(x \cos \theta + y \sin \theta) d\theta.
  \]

In practice, the projection data are measured at a discrete set of \(K\) angles:

\[
\{0, \Delta_\theta, 2\Delta_\theta, 3\Delta_\theta, \ldots, (K-1)\Delta_\theta\},
\]

where \(\Delta_\theta = \pi/K\). Likewise, the projection data will be sampled with a spatial sample spacing of \(\Delta_t\), so we will have measurements of \(p_\theta(t)\) only for the angles:

\[
\theta = 0, \Delta_\theta, 2\Delta_\theta, \ldots, (K-1)\Delta_\theta.
\]

\(^1\)For reference, consult Chapter 10 of the textbook, pages 321 through 326.
and spatial locations:

\[ t = -\frac{N}{2} \Delta t, \left( -\frac{N}{2} + 1 \right)\Delta t, \ldots, -\Delta t, 0, \Delta t, \ldots, \left( \frac{N}{2} - 2 \right)\Delta t, \left( \frac{N}{2} - 1 \right)\Delta t. \]

The course web page contains a link to several Matlab data files. After saving your file, the Matlab command `>> load filename`, where `filename` is the name of your data file, will load the Matlab variables `t`, `theta`, and `p_data`. The \(1 \times K\) array `t` is the spatial indices for the projections (in meters); the \(1 \times N\) array `theta` is the angular indices for the projections (in radians); and the \(N \times K\) array `p_data` is the projection data. The Matlab command:

`>> imagesc(theta, t, p_data);`

will provide a plot of the projection data.

Your first step of processing is to compute the Fourier transform for each angular projection. To do this, you need to apply the FFT to each column of the projection data array. Fortunately, the Matlab routine `fft` will do exactly this when applied to the projection data array `p_data`. The command to do this would be:

`P_data = fft(p_data)*dt`

where `dt` is the sample spacing.

Next, you will need to filter the data. To do this, you need to create an \(N \times K\) filter array whose columns contain the desired frequency response. (Hint: as we have done for other problems, you might use the `meshgrid` command to make an appropriate index array in angle and frequency, then create the required frequency response. Each column should contain the same values. Be sure to think about issues involving `fftshear` when doing this.) The following commands should form the filtered projections:

\[ G = P_data .* H; \]
\[ g = ifft(G)*N*df; \]

where `H` is your frequency response array and `df` is the sample spacing in frequency that results from the application of the `fft` algorithm.

The \(N \times K\) array `g` should contain the filtered projection data. Your final task is to carry out the backprojection step. To do this, you need to initialize an \(N \times N\) attenuation array, then fill in its values according to the backprojection equation:

\[ s(x, y) = \int_{0}^{\pi} g_\theta (x \cos \theta + y \sin \theta) d\theta. \]

Here is a suggested approach to doing this:
\begin{verbatim}
x1 = t;
x2 = t;
s = zeros(N,N);
for k=1:K
    for n1=1:N
        for n2=1:N
            t_k = x(n1)*cos(theta(k)) + y(n2)*sin(theta(k));
            index = ???;
            s(n1,n2) = s(n1,n2) + g(index, k)*dtheta;
        end;
    end;
end;
dtheta is the sample spacing for the angular variable $\theta$, and the variable index needs to be the index that is closest to selecting the value of $g_\theta(t)$ for $t = x \cos \theta + y \sin \theta$. You will need to figure out what Matlab statement should replace ??.
Your attenuation map $s(x, y)$ will look something like the one in the figure below.
\end{verbatim}

Like this one, your attenuation map will have a well-defined circular region within the oval region. You will need to determine the diameter (in cm) of the circular region. For instance, the circular region in this figure has a diameter of about 3.5 cm. Your report should include the following information:

(a) a reconstructed image of the attenuation map, with accurate labels of the spatial axes;

(b) a quantitative measurement of the diameter of the circular region of your attenuation map; and
(c) a description of the method you used, including a listing of any software you developed.

You may not work with other students on the development of your code.

2. (Doppler Processing) A Doppler radar system uses a pulse of the form:

\[ s(t) = \text{rect}\left(\frac{t - T/2}{T}\right) \cos(2\pi f_0 t), \]

with \( T = 250 \mu s \) and \( f_0 = 100 \text{ GHz} \) to monitor a moving object. Accordingly, the received signal is of the form:

\[ r(t) = \text{rect}\left(\frac{t - T/2 - 2d_0/c}{T}\right) \cos[2\pi f_0(1 + 2v/c)t - \phi], \]

where \( d_0 \) is the distance to the object, \( c \) is the speed of light, and \( v \) is the object’s velocity component toward the radar. (If \( v \) is negative, the object is moving away from the radar.)

Suppose this signal is processed as shown below:

\[ r(t) \]

\[ \cos(2\pi f_1 t) \]

\[ \times \]

\[ \text{LPF} \]

\[ r_I(t) \]

\[ r(t) \]

\[ -\sin(2\pi f_1 t) \]

\[ \times \]

\[ \text{LPF} \]

\[ r_Q(t) \]

to produce the complex-valued signal

\[ r(t) = r_I(t) + jr_Q(t), \]

where the local oscillator has frequency \( f_1 = f_0 - f_{\text{error}} \), where \( f_{\text{error}} = 15 \text{ kHz} \). Keep in mind that the relationship between your signal data and the object’s velocity will be slightly different than it was in our previous project because the oscillator frequency is mismatched with the transmitter frequency.
The project web page contains links to data files for each student in the class. The files are Matlab data files, with the '.mat' extension. Upon saving this file, the matlab command `load <filename>` will load the variables t and r which correspond to the time samples and corresponding signal samples for your data.

Your objective is to process these data to determine the velocity of your target. Be sure to consider the fact that the received signal was mixed with a local oscillator with a frequency different from \( f_0 \). You might find the following identity to be useful:

\[
\cos(A) \sin(B) = 0.5 \sin(A + B) - 0.5 \sin(A - B).
\]

You should turn in a professional report indicating the velocity of your target in kilometers per hour along with any plots and equations that demonstrate the method by which you computed your answer.

You may not work with other students on the development of your code.

3. (Spectroscopy) Light of unknown wavelength illuminates a screen with two holes that are separated by a distance of 0.05 mm. The diameter of each hole is 5 µm. The intensity for the field that propagates from this screen is detected by a charge-coupled-device detector at a distance of 10 cm. Recall that the intensity for a monochromatic field \( s_d(x, y) \) is defined as \( I_d(x, y) = |s_d(x, y)|^2 \).

The exam web page contains links to data files for each student in the class. The files are Matlab data files, with the '.mat' extension. Upon saving your file, the matlab command `load <filename>` will load the variables D, d, sep, x_d, y_d, and I_d. The variables D, d, and sep correspond to the pin-hole diameters, the distance from the screen to the detector, and the pin-hole separation, respectively. The variables x_d, y_d, and I_d correspond to the spatial axes and the detector intensity. Your source contains light of one, two, or three different wavelengths. Your goal is to process the intensity data to determine a) the number of distinct wavelengths at which your source has energy, and b) the relative intensities of the light at these wavelengths. For instance, your answer might be of the form:

The light in my signal contains energy at 0.4 µm and 0.6 µm, and the intensity of light at 0.4 µm is 3.2 times as large as the intensity of light at 0.6 µm.

Your report should also include supporting materials such as a description of your method, and a listing of your matlab code.

You may not work with other students on the development of your code.