Assignment 5

General Info

- The work can be done anywhere where MATLAB and the related toolboxes are available.
- A **written report** is required. The report is free form but should include results, figures, and any code you wrote, as well as discussions of what you observe.
- Reports should be submitted as an electronic copy in **PDF** format on **Canvas** before the due date.
- Late submissions will not be accepted.
- Better documentation and clearer discussions of your work improves our ability to fairly mark your report. Make sure your report is well structured, organized, and clear. Your report has to follow the order of questions. Give all relevant code right before the results and discussion of each part, not in an appendix.

1. Reconstruction from Projections

In this question, we will be exploring different interpolators and filters that can be employed to enhance the 2D reconstruction from linear projections. We will be using the Signal-to-Noise Ratio and Mean Squared Error similarity metrics to evaluate the effectiveness of each parameter. The `iradon` function in MATLAB automatically includes the following filters and interpolators which we will be investigating.

```
interpolators = {'nearest';'linear';'spline';'pchip';'cubic';'v5cubic'};
filters = {'none';'Ram-Lak';'Shepp-Logan';'Cosine';'Hamming';'Hann'};
```

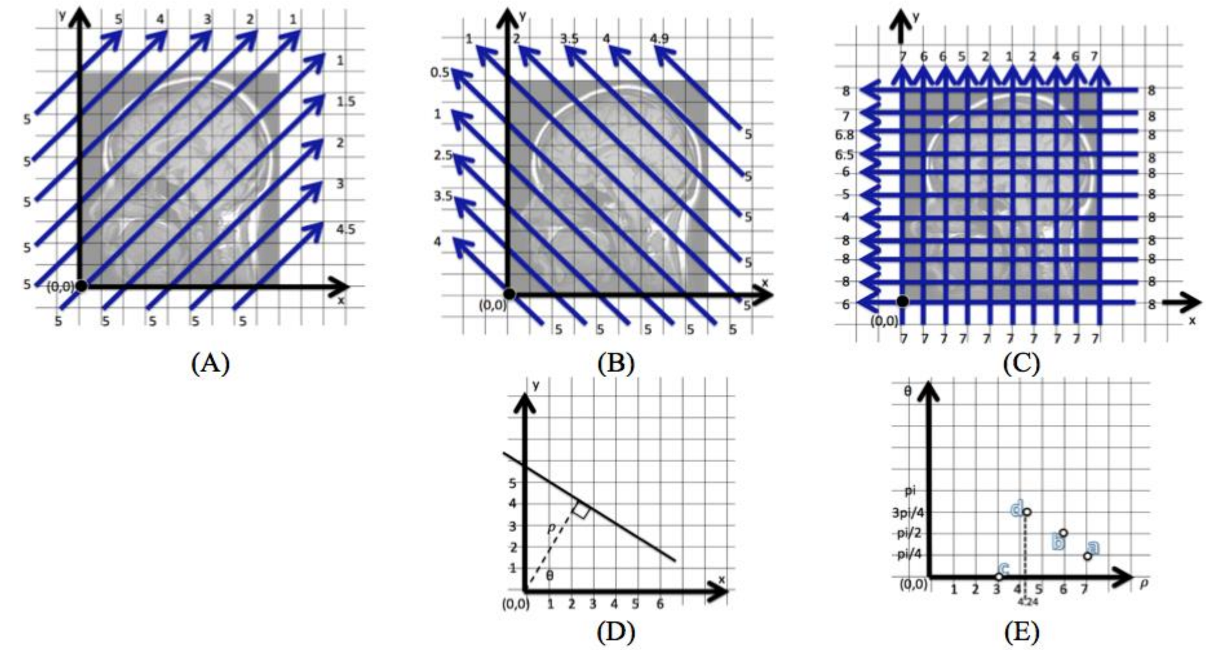
Load the Q1_student.m file and perform the following:

(a) Run the code and comment on the results generated by default.
(b) Try all pairwise-combinations of the parameters listed above. Which pair of parameters produces the best results in terms of a low MSE and a high SNR? (Hint: use loops to automate the process!) Does this correspond with what you visually observe to be the best?
(c) Change the number of angles from 200 to 20. Are the parameters chosen for part b still the best candidates? Why or why not?

2. Computed Tomography

Figures (A-C) represent x-ray beams passing through a slice of brain tissue. The numbers at the tail and head of each arrow indicate the intensities of the x-ray beam at the source and detector, respectively. The grid lines are represented by 1 unit. The relationship between the signal strength at the transmitter I_0 and signal strength at the receiver I is defined by the following equation:

$$I = I_0 \exp\left(-\int \mu(x,y)ds\right)$$



(a) What does μ represent in this application?
(b) What is the relationship between the input and the output signal strength and the radon transform?
(c) Figure (D) shows how a line in x-y space is represented using ρ and θ. Your goal is to report the Radon transform values at the 4 indicated points (a-d) in Figure (E), e.g. point d is at (ρ, θ) = $(3\sqrt{2}, 3\pi/4)$

3. JPEG Compression

JPEG compression is a commonly used method of lossy compression for digital images. The degree of compression can be adjusted, allowing a selectable trade-off between storage size and image quality.

The JPEG encoding process has the following 5 steps:

1. The representation of the colors in the image is converted from RGB to YC_BC_R.
2. The resolution of the chroma data is reduced, usually by a factor of 2.
3. The image is split into blocks of 8×8 pixels, and for each block, each of the Y, C_B, and C_R data undergoes the Discrete Cosine Transform (DCT). A DCT is similar to a Fourier transform in the sense that it produces a kind of spatial frequency spectrum.
4. The amplitudes of the frequency components are quantized.
5. The resulting data for all 8×8 blocks is further compressed with a lossless algorithm, a variant of Huffman encoding.

In a previous assignment question ("Redundancy in the Color Space"), we observed the effects of subsampling the color space (steps 1 and 2 in the above list of JPEG encoding steps). In this problem we will be focusing on steps 3 and 4. For the sake of simplicity we will perform JPEG compression on a scalar gray scale image from the default MATLAB path: `cameraman.tif`.

(a) Use MATLAB's `dctmtx` function to generate an 8x8 DCT transformation matrix. Plot this matrix using `imshow` and briefly explain what the matrix does.

(b) Prior to performing DCT on cameraman, the intensity values must be shifted from a positive range of [0,255] to one centered around zero [-128,127] by subtracting 128 from every intensity value. Explain why.

(c) MATLAB's documentation for `dctmtx` mentions that: "For example, in JPEG compression, the DCT of each 8-by-8 block is computed. To perform this computation, use `dctmtx` to determine D, and then calculate each DCT using D*A*D' (where A is each 8-by-8 block)." To perform DCT on every 8x8 block of the image, use MATLAB's `blockproc` function. Plot the resulting transformed image and explain the results.

(d) To quantize the transformed image, the following matrix is used:

```
Q =
     1     1     1     2     2     2     4     4
     1     1     2     2     2     4     4     4
     1     2     2     2     4     4     4     8
     2     2     2     4     4     4     8     8
     2     2     4     4     4     8     8     8
     2     4     4     4     8     8     8    16
     4     4     4     8     8     8    16    16
     4     4     8     8     8    16    16    16
```

And the quantized DCT coefficients are computed as:

```
B = round(G./(q_level*Q));
```

Where *G* is the un-quantized DCT coefficients; *Q* is the quantization matrix; *q_level* is the level of quantization we want to achieve; and *B* is the resulting quantized (normalized) coefficients. To achieve a higher level of quantization, *G* is divided by integer multiples of *Q*. You will be asked to perform this quantization in the next question. For this part, only explain why this quantization matrix has these values.

4. JPEG Decoding

We have now arrived at step 5 of JPEG Compression. Before talking about lossless data compression, we will try to verify the effects of quantization.

(a) Perform the quantization as described in Q3(d). As we divide each block by the quantization matrix and round it, we are increasing the sparsity of how we represent the image. Effectively, reducing many values of *G* to zero. To observe this, plot the histogram of *G* using the `hist` function and compare it to the histogram of *B* with *q_level*=10. Briefly comment on the histograms.

(b) Now that we have quantized and compressed our image, it is time to reconstruct it. The first step is to un-normalize our coefficients. To do this, we multiply *B* by *Q* the same number of times that we divided it in the quantization step to obtain a new *G*.

```
G = B.*(q_level*Q);
```

We then perform an inverse block-wise DCT. The forward DCT transform was defined as G=D*A*D', the inverse DCT transform is A=D'*G*D. Plot the reconstructed image as shown below and explain the differences compared to the original image. Hint: remember to add back the 128 and round the image values to the range of [0 255].

(c) Perform JPEG compression at different levels of quantization and explain your observations. What happens at very high q_level values? Zoom in on the camera and explain the artefacts that you observe as a result of this quantization.

5. Lossless (Huffman) Encoding

In this problem we will focus on step 5 of the JPEG encoding process. The JPEG compression standard uses two forms of lossless compression, run-length and Huffman, to further compress the quantized DCT coefficients. Let's explore the effects of Huffman coding on the cameraman image.

(a) Briefly explain information entropy and its relationship to compression.
(b) Compute the entropy of the image and calculate the maximum compression that can be expected?
(c) Will this be obtained?

6. [BONUS] Predictive Coding

Predictive coding is another lossless compression technique which allows one to compress an image beyond the limits of Huffman coding. A simple way to employ predictive coding on an image is by storing the difference between adjacent rows in an image rather than the pixel values such that

$$e(x,y) = f(x,y) - \hat{f}(x,y)$$

$$\hat{f}(x,y) = \text{round}[\alpha f(x,y-1)], \alpha = 1.$$

(a) Encode the cameraman image using the scheme shown above and plot the resulting image. What does this image look like?
(b) To see why this simple method can produce better compression, plot the histogram of the camera image before and after the encoding step. Comment on the differences between the two. Calculate the entropy of the two histograms to quantify the difference between potential compression ratios that can be achieved.
(c) Reconstruct the encoded the image and calculate the error between the original and reconstructed versions. Is this method truly lossless, why or why not?

End of assignment 5