Grant Roberts

MP710: Advanced MRI

10/18/2018

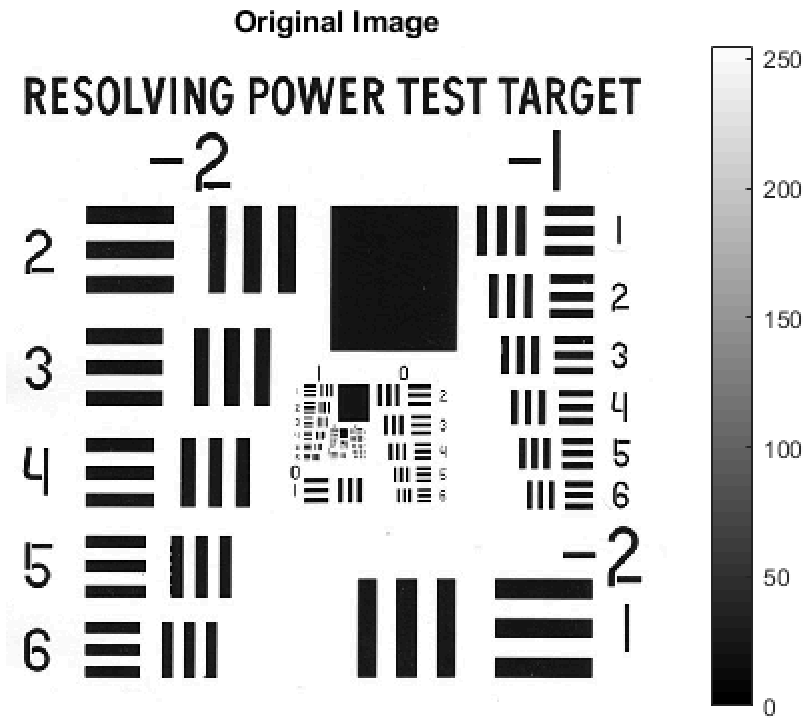
Homework 3

Problem 1

Part a)

Generate the corresponding k-space (aka Fourier space) data and provide images with magnitude and phase of those k-space data. Those data will function as our simulated 'acquired k-space data' from now on.

```
%Got to HW3_Files directory
%Read in file and show image
file1 = [pwd '\test_pattern.png'];
data = importdata(file1);
figure; imshow(data,[]); title('Original Image'); truesize; colorbar
```

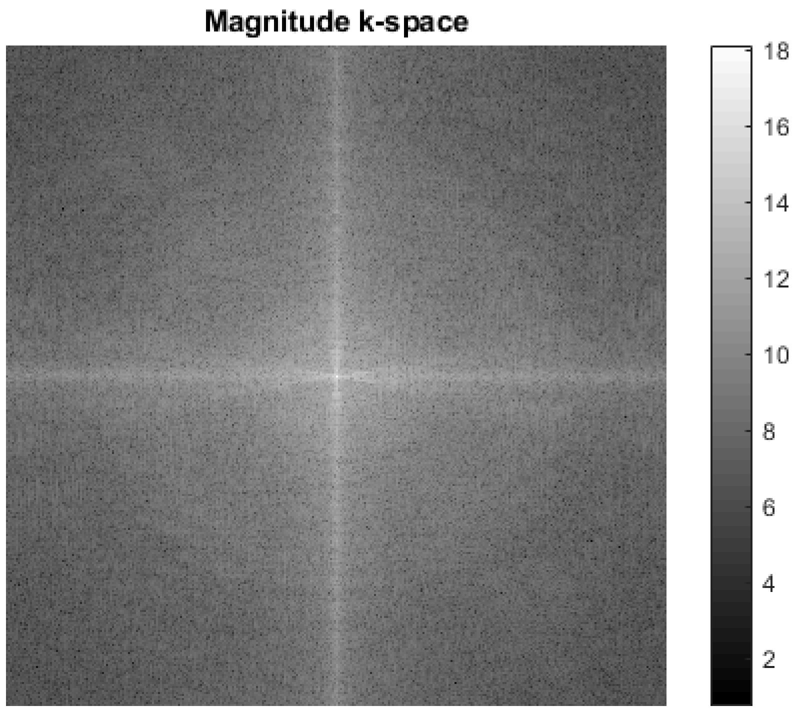


```
Warning: Image is too big to fit on screen; displaying at 67%
Warning: Image is too big to fit on screen; displaying at 67%
```

```
%Perform Fourier Transform (fast fourier)
DATA = fftshift(fft2(data));

    %Can we get back to the original image?
    data_test = uint8(ifft2(ifftshift(DATA)));
    test = isequal(data,data_test);
    disp(test)
```

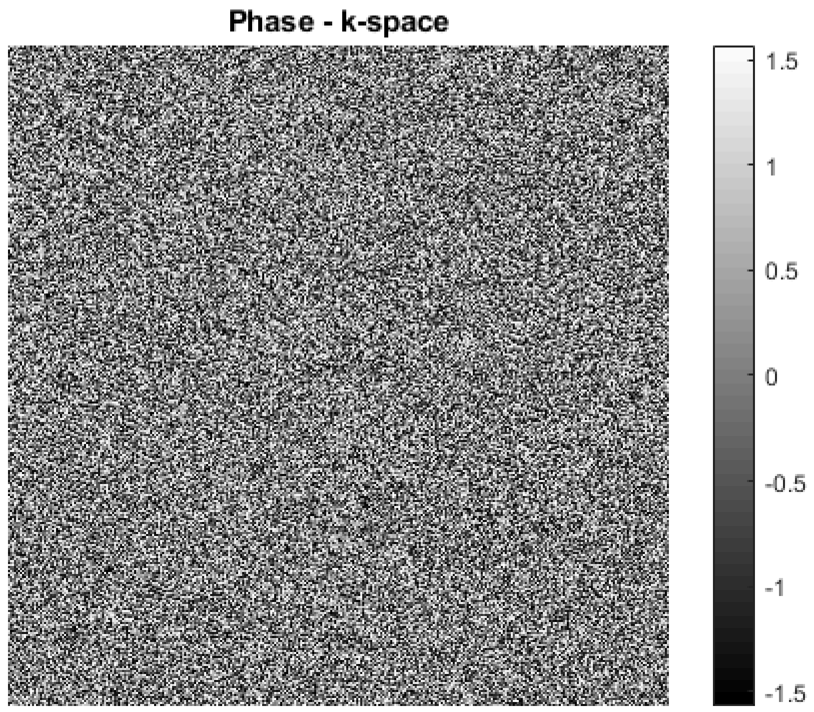
```
1
```

```
    % If test = 1, then we got back the original image

%Show magnitude and phase of k-space data
MAG = get_mag(DATA);
figure; imshow(log(MAG),[]); title('Magnitude k-space'); truesize; colorbar
```



```
Warning: Image is too big to fit on screen; displaying at 67%
Warning: Image is too big to fit on screen; displaying at 67%
```

```
PHASE = get_phase(DATA);
figure; imshow(PHASE,[]); title('Phase - k-space'); truesize; colorbar
```



```
Warning: Image is too big to fit on screen; displaying at 67%
Warning: Image is too big to fit on screen; displaying at 67%
```

Part b)

Simulate a 'partial Fourier' acquisition that samples 50% of the phase encodes by using only the central lines (rows) of k-space.

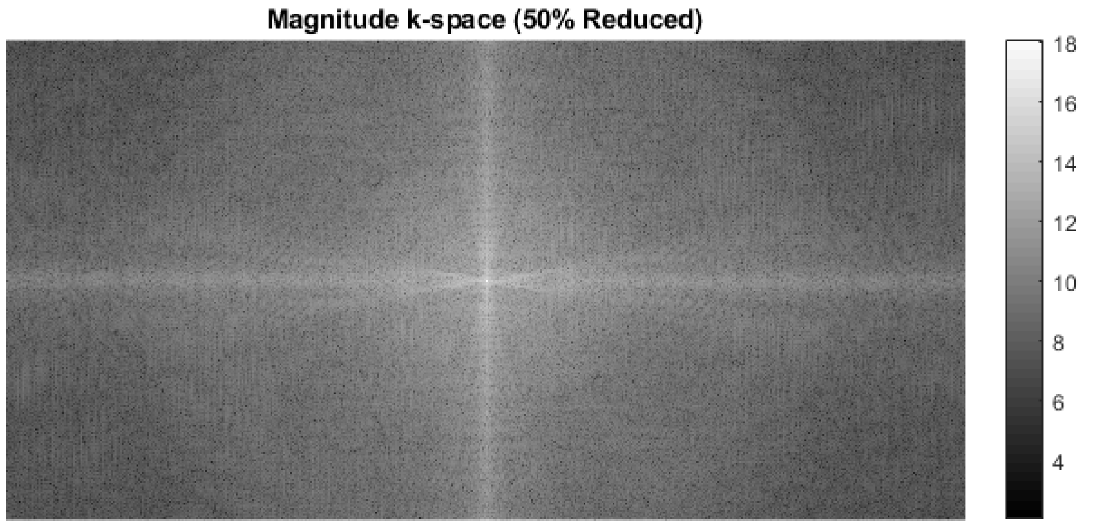
```
[Ny,Nx] = size(DATA);

index = (Ny*0.5)/2;
%Cut the data in half
PARTIAL = DATA((index+1:end-index),:);
%Looking at 1 column of data, samples should now be cut in half in the phase direction.
new_Ny = length(nonzeros(PARTIAL(:,1)))
```

```
new_Ny = 300
```

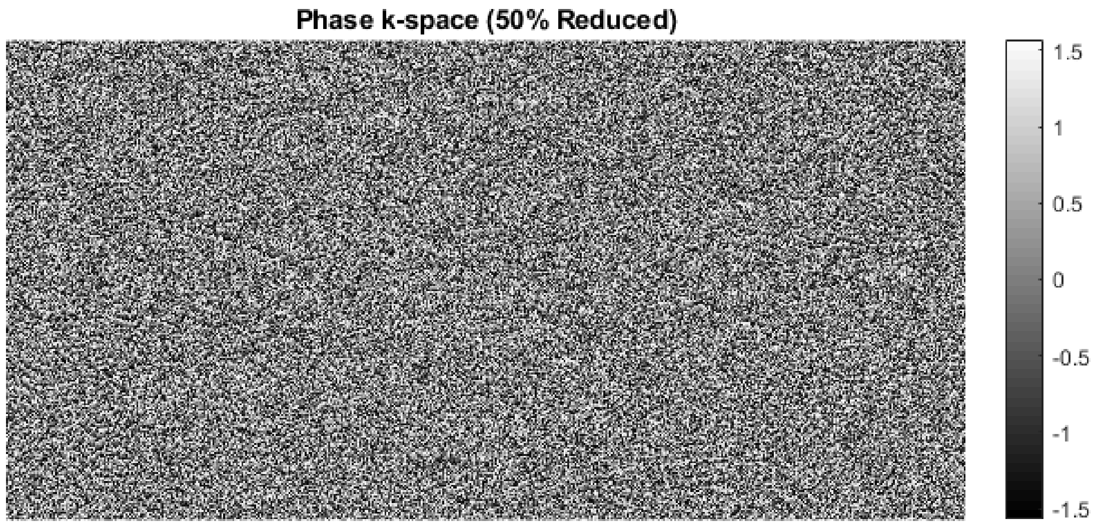
Provide images of the k-space data, and the resulting image (magnitude and phase) when reconstructed from this subset.

```
%Show magnitude of reduced k-space data
```

```
figure; imshow(log(get_mag(PARTIAL)),[]); title('Magnitude k-space (50% Reduced)'); truesize; c
```

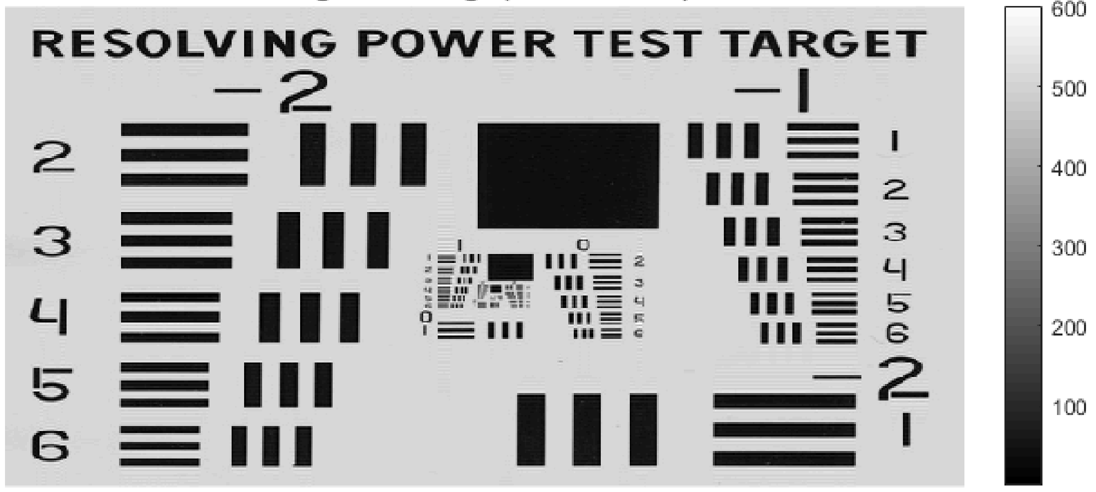


```
figure; imshow(get_phase(PARTIAL),[]); title('Phase k-space (50% Reduced)'); truesize; colorbar
```



```
%Reconstruct and show images
partial = ifft2(PARTIAL);
mag1b = get_mag(partial);
figure; imshow(mag1b,[]); title('Magnitude Image (50% Reduced)'); truesize; colorbar
```

Magnitude Image (50% Reduced)



```
phase1b = get_phase(partial);
figure; imshow(phase1b,[]); title('Phase - Image Space (50% Reduced)'); truesize; colorbar
```

Phase - Image Space (50% Reduced)

1.5
1
0.5
0
-0.5
-1
-1.5

What position in the Matlab matrix represents the DC component of this frequency space representation?

The center of k-space represents the DC component. This is because the (demodulated) signal is unvarying in frequency. This zero-frequency signal corresponds to a fixed value that is equal to the integral of the signal in the image domain over all space. Most of the k-space "energy" exists at and around this point.

```
%Where is the DC component where most of the signal is contained.
[maxX,iX] = max(max(DATA,[],1));
[maxY,iY] = max(max(DATA,[],2));
formatSpec = 'The DC component is located at [%d,%d].';
sprintf(formatSpec,iX,iY)
```

```
ans =
'The DC component is located at [301,301].'
```

How long would this acquisition have taken in comparison to the fully sampled data in a)?

This acquisition would have taken exactly half the time, because the phase encoding steps are halved.

Comment on differences of the phantom appearance in a) and b) in the image domain.

The phantom in part a) seems to have better resolution in the y-dimension. Zoomed in images of both will be shown below.

Image a

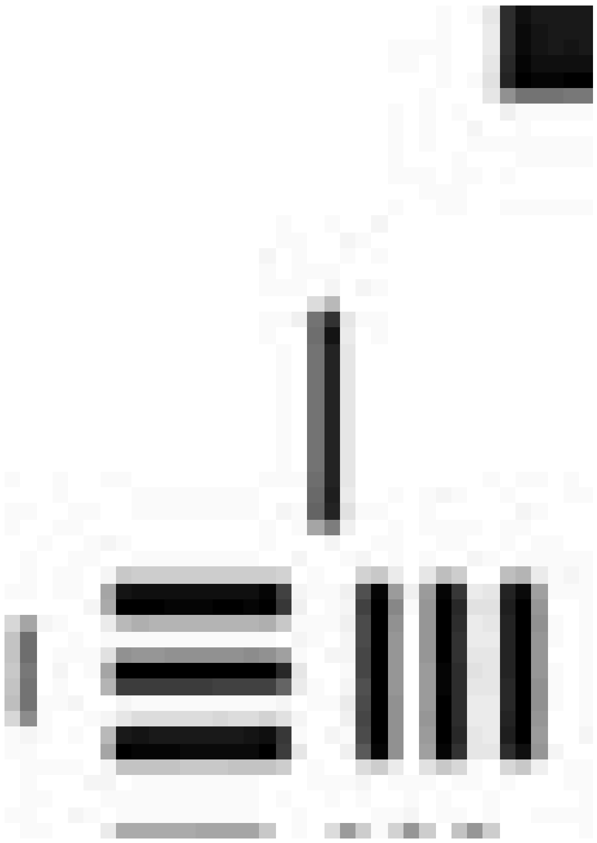
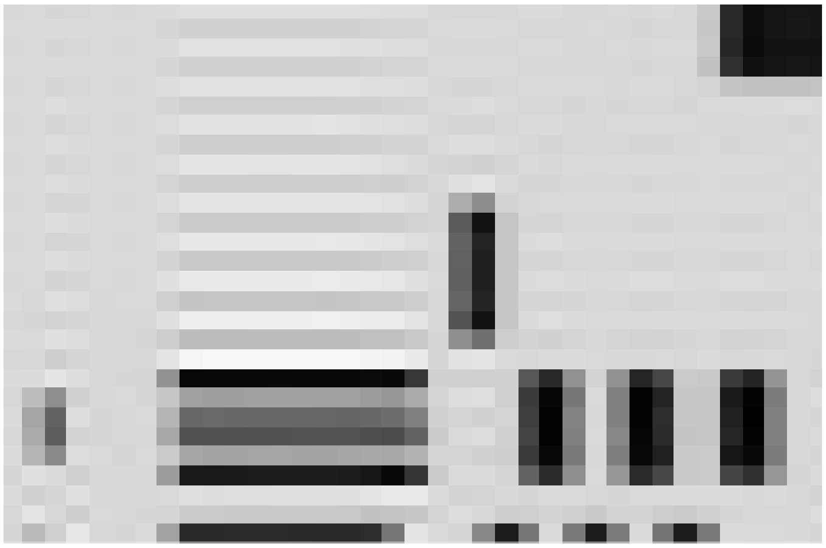


Image b



It is also evident that there is a strong Gibbs ringing artifact in the image from part b). Secondly, the background color is not white, as is was in part a). This is likely because the ringing pattern is creating high intensity signal (Gibbs overshoot), which is making the background not the highest intensity pixel value as it was in the original image.

Part c)

You will notice that the aspect ratio in the image in b) is different from a).

Explain why this occurs and demonstrate how that improper display can be fixed. Provide an image with proper aspect ratio (magnitude image is sufficient).

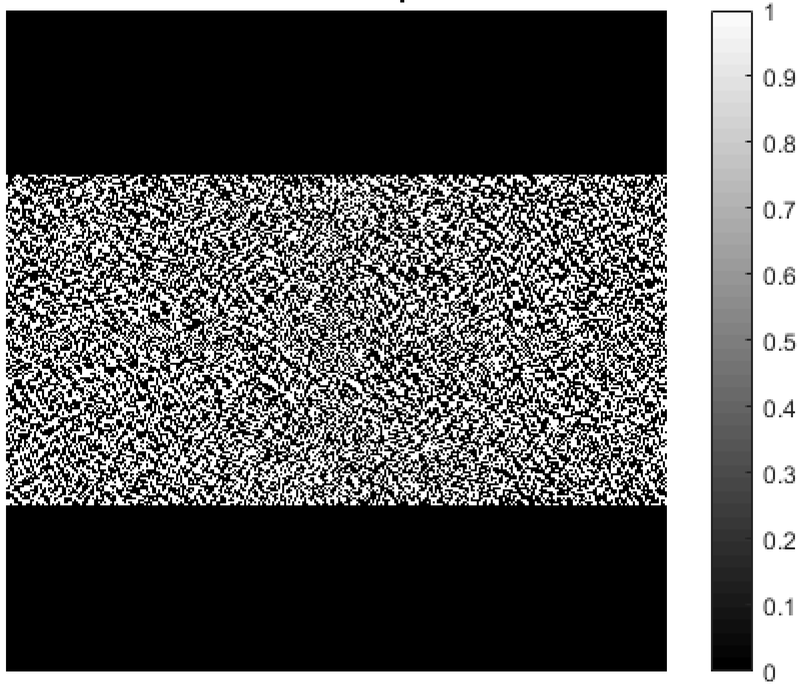
The aspect ratio is diminished because reconstructing k-space data into the image domain will preserve the matrix size (aspect ratio). In our case, since we halved the matrix, the aspect ratio in image b) will also be halved.

To fix this, we could simply zero fill out to the edge of k-space to preserve the aspect ratio and increase "apparent" resolution of the image.

```
%Pad the partial data set with zeros out to edges (creating a 600x600 matrix)
FULL = zeros(600);
FULL(index+1:end-index,:) = PARTIAL;
figure; imshow(FULL); title('Zero-filled k-space'); truesize; colorbar
```

```
Warning: Displaying real part of complex input.
```

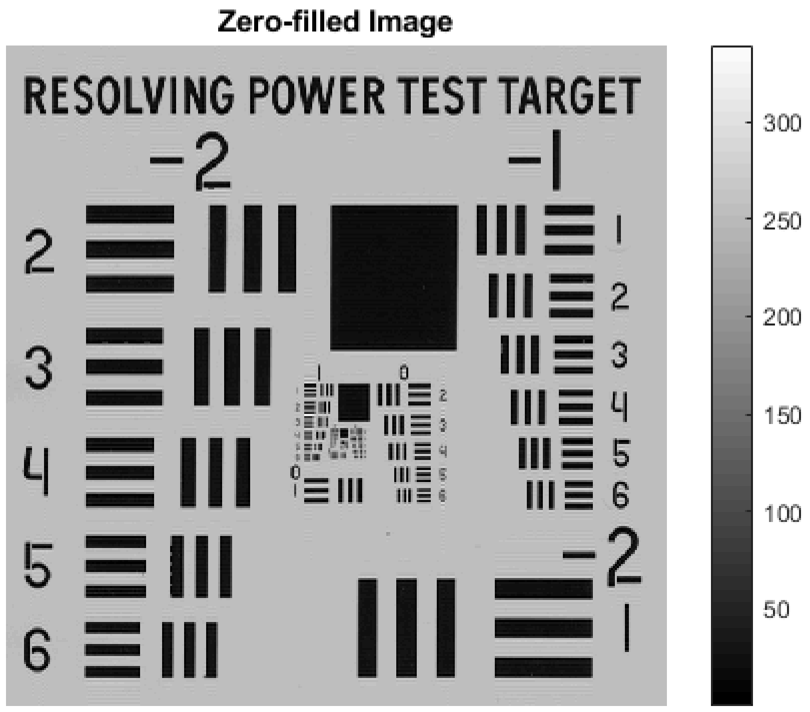
Zero-filled k-space



```
Warning: Image is too big to fit on screen; displaying at 67%
Warning: Image is too big to fit on screen; displaying at 67%
```

```
%Reconstruct and show
full = ifft2(FULL);
mag1c = get_mag(full);
```

```
figure; imshow(mag1c,[]); title('Zero-filled Image'); truesize; colorbar
```



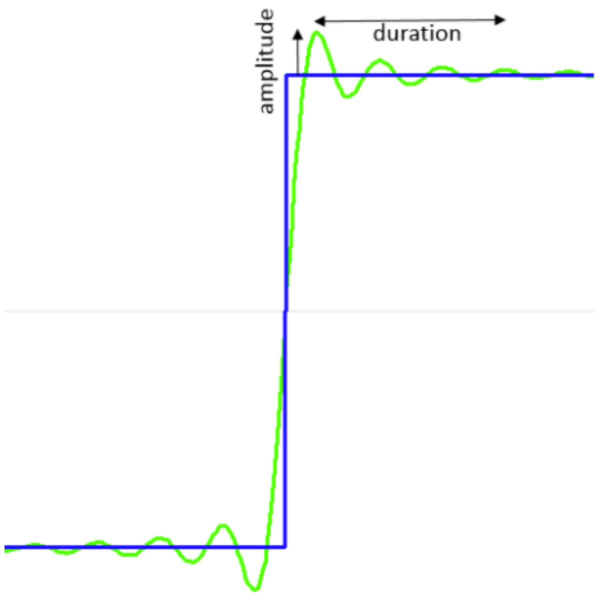
```
Warning: Image is too big to fit on screen; displaying at 67%
Warning: Image is too big to fit on screen; displaying at 67%
```

Part d)

Your clinical collaborator tells you that he has observed artefacts in your image in b) that are unacceptable for him: high contrast edges are replicated with offsets to the true spatial location of those edges.

Explain the origin of this artefact.

This artefact is termed "Gibbs ringing" and refers to the fact that high frequency signals are necessary to produce sharply changing signal intensities (high contrast edges) in image space. If we imagine a rect function, we know that the higher frequency sinusoidal waves are required to produce better representations of a true rectangle. An example of a lower frequency representation of a half-rect function is shown below:



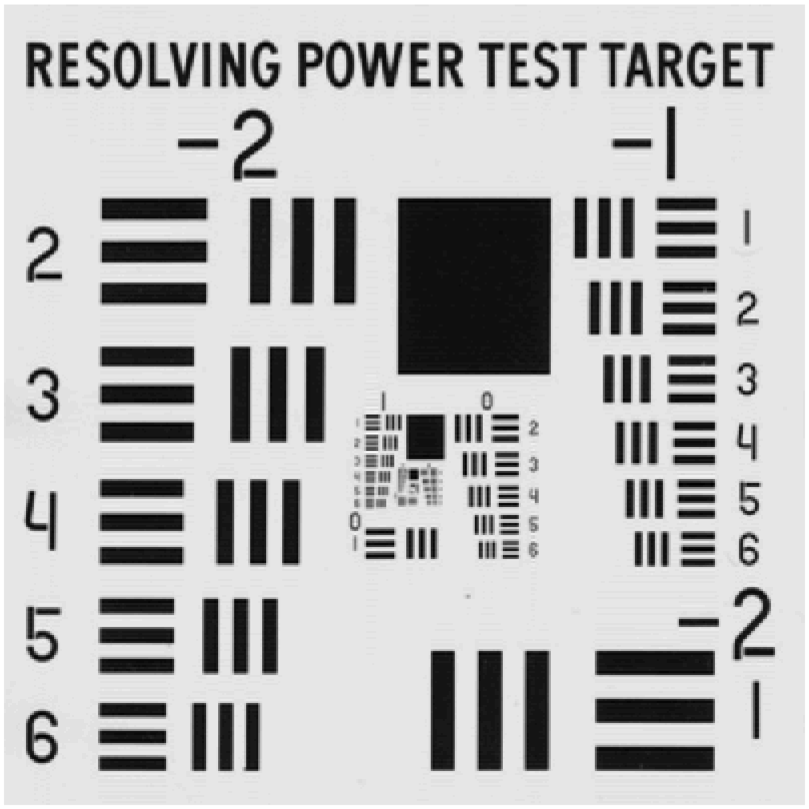
If high frequency signals are excluded, as they were in the above example, sinusoidal components of the low frequency waves will continue past the true edge of the object and produce repeating signals past that edge (ringing).

Implement a remedy to reduce this artefact. You can only use the data you have from 1b for this problem. In other words, you do not have the fully sampled data set. Provide an image of your result and comment on possible drawbacks of your artefact reduction approach.

One potential fix would be to use an apodizing function in the k-space domain. For this, I will multiply the k-space domain by a Hanning window, which will taper off the increasing spatial frequency towards the edge of k-space (instead of abruptly becoming 0 at 300). On that note, I will continue to use the zero-padded 600x600 matrix to improve image quality.

```
%download 2D filtering window @ https://www.mathworks.com/matlabcentral/fileexchange/43827-two-
filter = window2(600,600,@hanning);

FILT = filter.*FULL;
filt = ifft2(FILT);
figure; imshow(abs(filt),[]); title('Apodized Image')      % Apodized image
```

Apodized Image



```
Warning: Image is too big to fit on screen; displaying at 67%
```

Doing this removed the ringing artifact but blurred the image slightly, degrading resolution of fine edges.

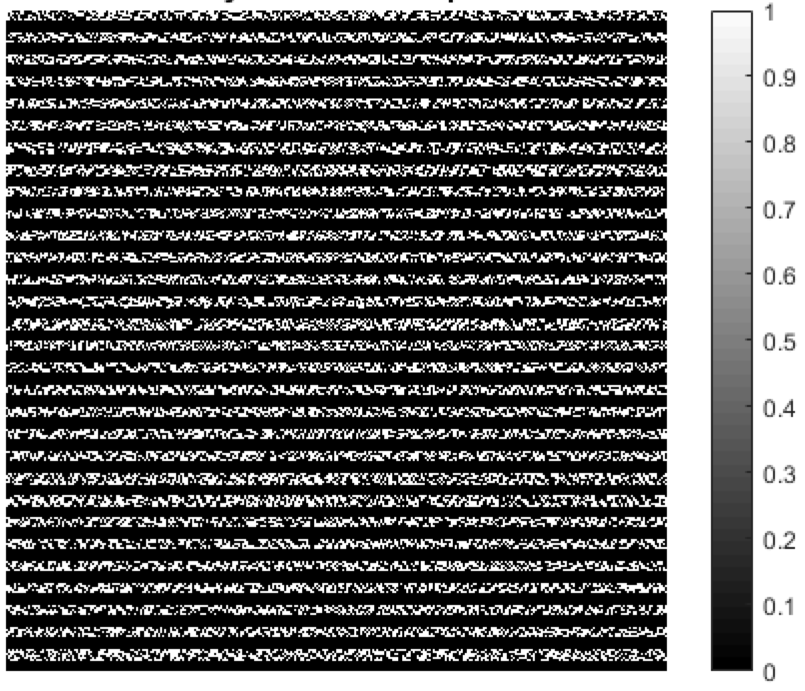
Part e)

Now simulate an alternative approach for sampling fewer k-space lines: only use every other line (row).

```
%Sample every other line from the original k-space data
OTHER = zeros(600);
for i = 1:2:600
    OTHER(i,:) = DATA(i,:);
end
imshow(OTHER); title('Every Other Line k-space'); truesize; colorbar
```

```
Warning: Displaying real part of complex input.
```

Every Other Line k-space

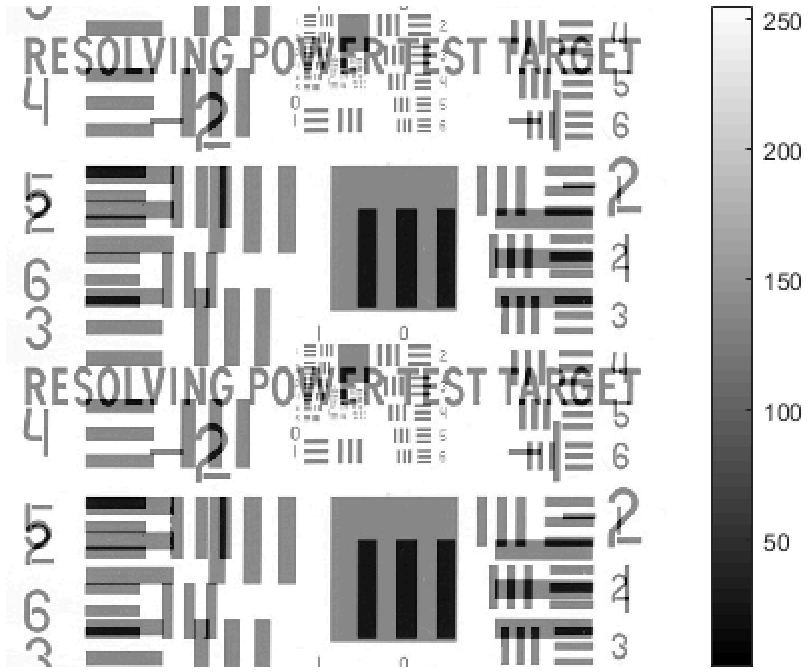


```
Warning: Image is too big to fit on screen; displaying at 67%
Warning: Image is too big to fit on screen; displaying at 67%
```

Provide the image reconstructed from this k-space subset (magnitude image is sufficient)

```
%Reconstruct the partially sampled dataset amd show
other = ifft2(OTHER);
mag1e = get_mag(other);
imshow(mag1e,[]); title('Aliased Magnitude Image'); truesize; colorbar
```

Aliased Magnitude Image



```
Warning: Image is too big to fit on screen; displaying at 67%
Warning: Image is too big to fit on screen; displaying at 67%
```

Comment on the appearance of the resulting image and the origin of any artefacts.

This image is heavily aliased (wrap-around artifact) in the y-direction. This is because our k-space spacing Δk_y has increased, and thus our field of view in image space has decreased, resulting in higher frequency signals being misinterpreted as low frequency (causing wrap-around).

Problem 2

Now we will work with data acquired on a clinical 1.5T system, in this case a scanner from GE Healthcare. A phantom was placed in the scanner and a single slice was imaged with a T1-weighted sequence (acquired to emphasize differences between tissues in T1 relaxation). The imaging parameters were as follows: # of samples per readout = 512, # of phase encodes = 512, FOV: 18 cm x 18 cm, receiver bandwidth = 20.83 kHz, slice thickness = 2 mm, repetition time TR = 425 ms, echo time TE = 17 ms. The data were acquired with a single channel head coil (birdcage design). The data are stored as recorded by the scanner in a binary file that I renamed to RAW_512_512.mri. That files has a vendor specific format including a header with lots of detailed information on the scan parameters and some space reserved for calibration data which are no longer used. I also provided a Matlab routine called read_raw.m. This routine can be used to read the acquired data into a complex matrix, where each row represents an acquired echo. Type >>help read_raw to see how the routine is called properly.

```
B0 = 1.5;               % Field Strength (T)
gammaBar = 42.58*10^6   % Gyromagnetic ratio (Hz/T)
```

```
gammaBar = 42580000
```

```
f0 = B0/gammaBar;       % Primary precession frequency (Hz)
TR = 0.435;             % Repetition time (s)
TE = 0.017;             % Echo time (s)
Npe = 512;              % Number phase encodes
Nfe = 512;              % Number frequency encodes
FOVx = 0.18;            % Field of view in x-dimension (m)
FOVy = 0.18;            % Field of view in y-dimension (m)
BWr = 20.83*10^3        % Receiver bandwidth (Hz)
```

```
BWr = 20830
```

```
deltaZ = 0.002;         % Slice thickness (m)
```

Part a)

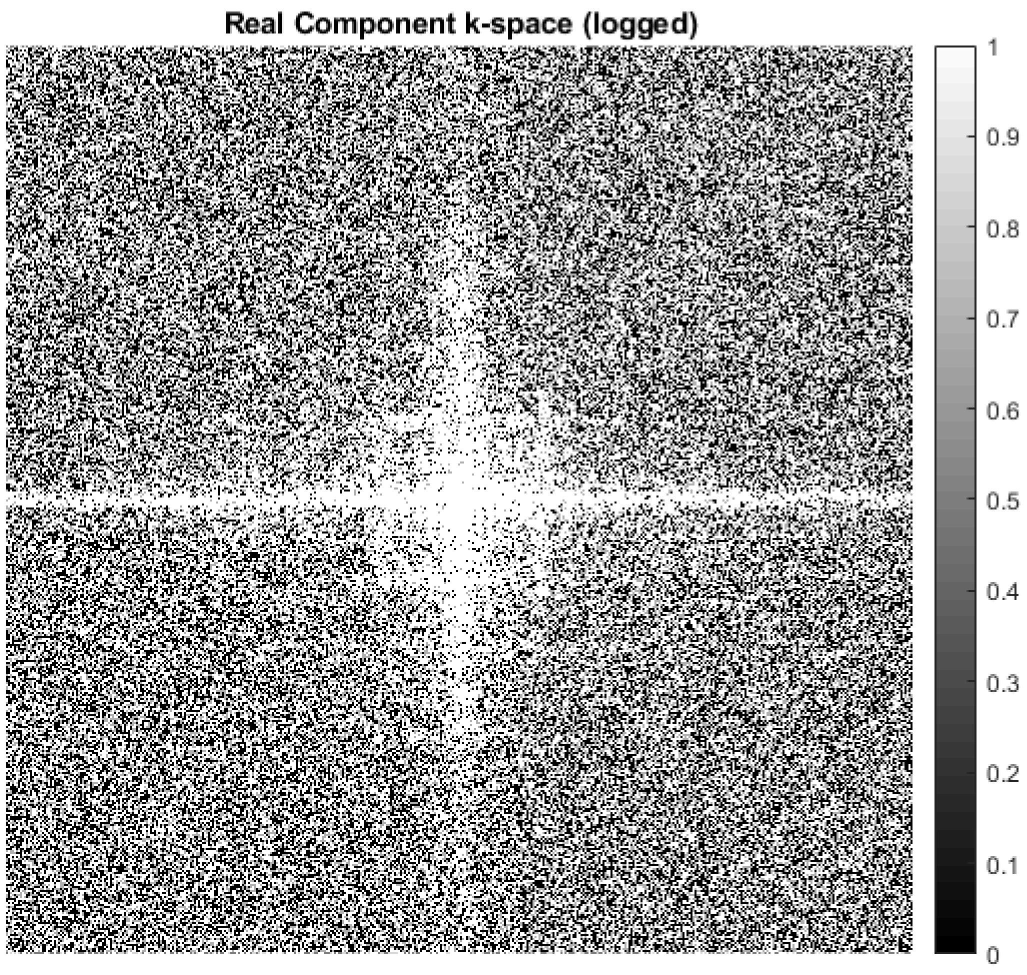
Use read_raw to read the raw data file RAW_512_512.mri into a Matlab matrix.

```
PHANT = read_raw('RAW_512_512.mri',512,512,1);
```

Provide images that show the magnitude, phase, the real, and the imaginary components of the data acquired in Fourier space. Use a logarithmic scale for all of these images except of for the phase distribution.

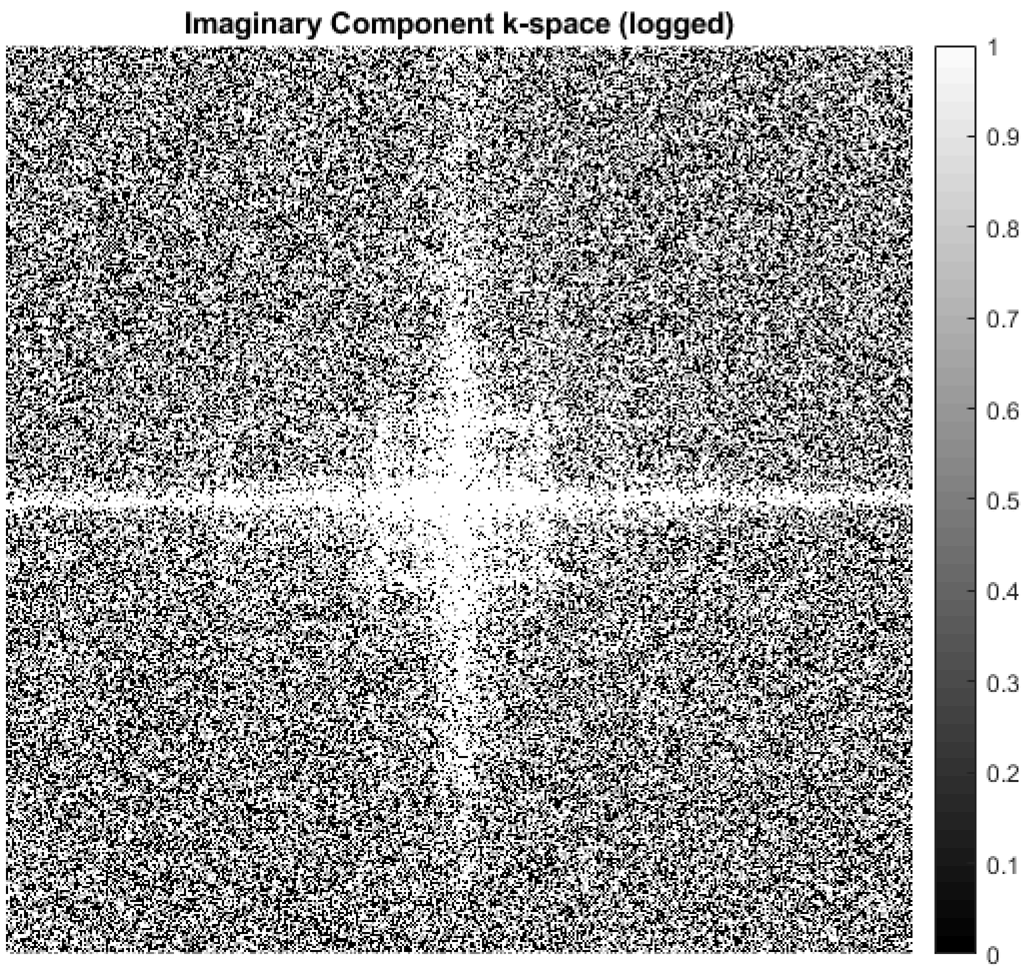
```
%Show real, imaginary, magnitude, and phase components of the original k-space dataset.
figure; imshow(log(real(PHANT))); title('Real Component k-space (logged)'); truesize; colorbar
```

```
Warning: Displaying real part of complex input.
```

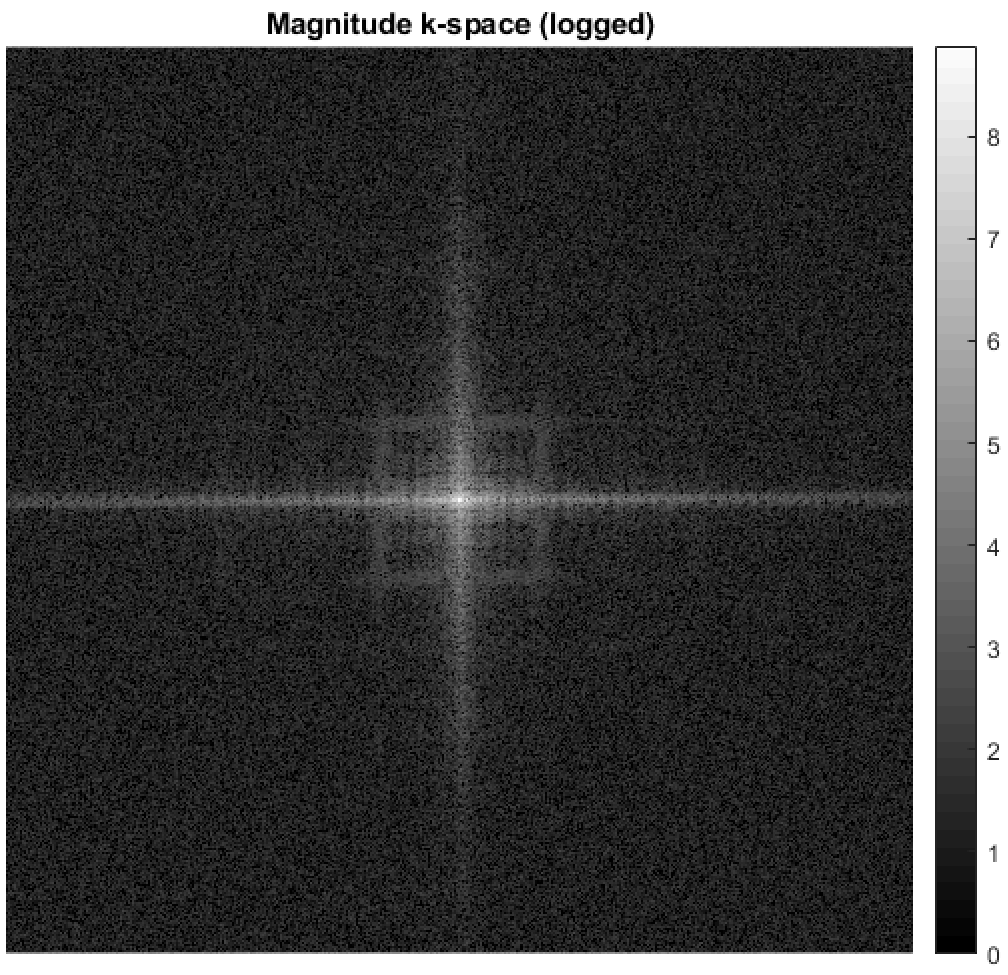


```
figure; imshow(log(imag(PHANT))); title('Imaginary Component k-space (logged)'); truesize; colo
```

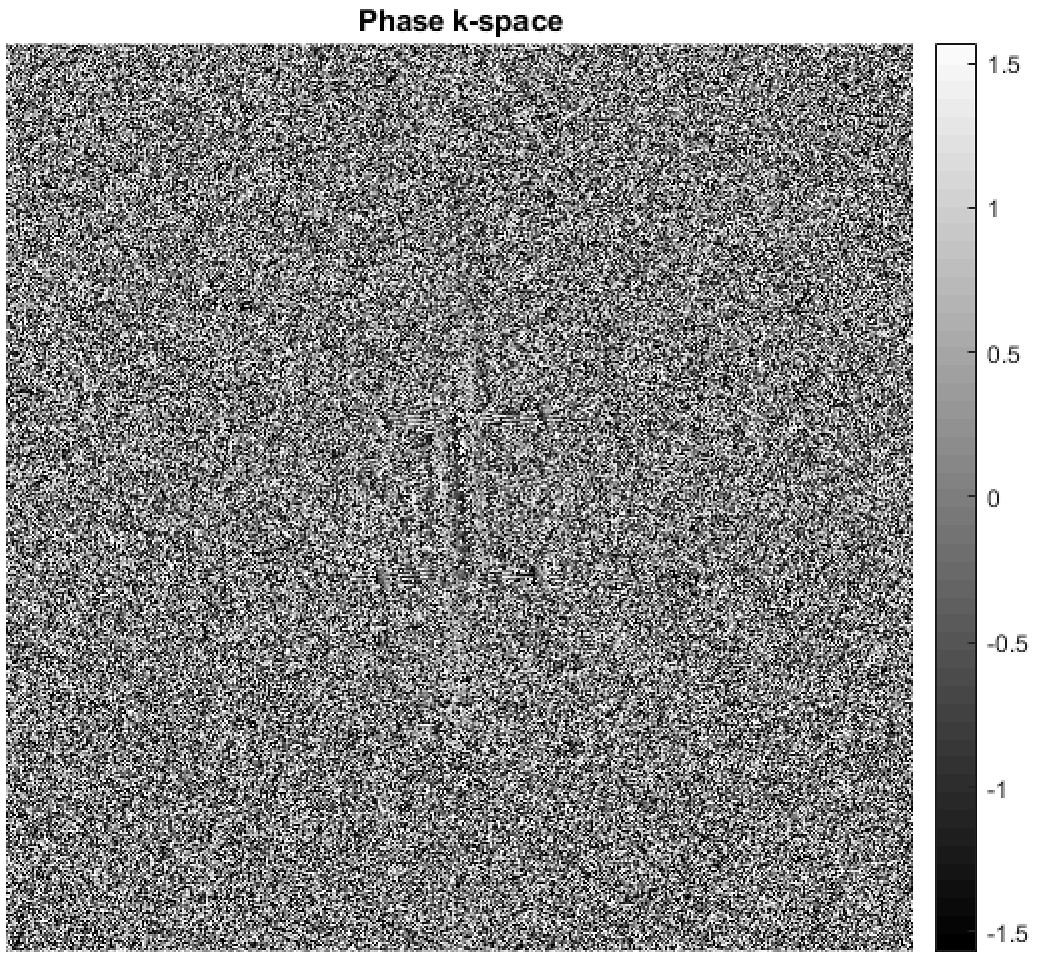
```
Warning: Displaying real part of complex input.
```



```
figure; imshow(log(get_mag(PHANT)),[]); title('Magnitude k-space (logged)'); truesize; colorbar
```



```
figure; imshow(get_phase(PHANT),[]); title('Phase k-space'); truesize; colorbar
```



Comment on similarities or differences of these k-space data to the ones numerically generated in problem 1.

In terms of similarities, in both images there are two bright streaks running along the kx and ky axis, indicating high amounts of low frequency in the x- and y-dimension of the iamge. Also, there is a bright center in both k-space datasets as expected, since this relates to the DC component of the signal.

In terms of differences, the k-space image from problem 1 (Image 1) has more signal spread out in k-space whereas the image above (Image 2) has most of the k-space signal in the center of k-space. Secondly, in Image 2, there seems to a square appearance of bright signal near the center of k-space, indicating that a strong component of the image is varying low spatial frequency at a constant ky (top and bottom of square) and constant kx (sides of square). In image 1, there were "spokes" extending out from the center of k-space, which is not seen in Image 2.

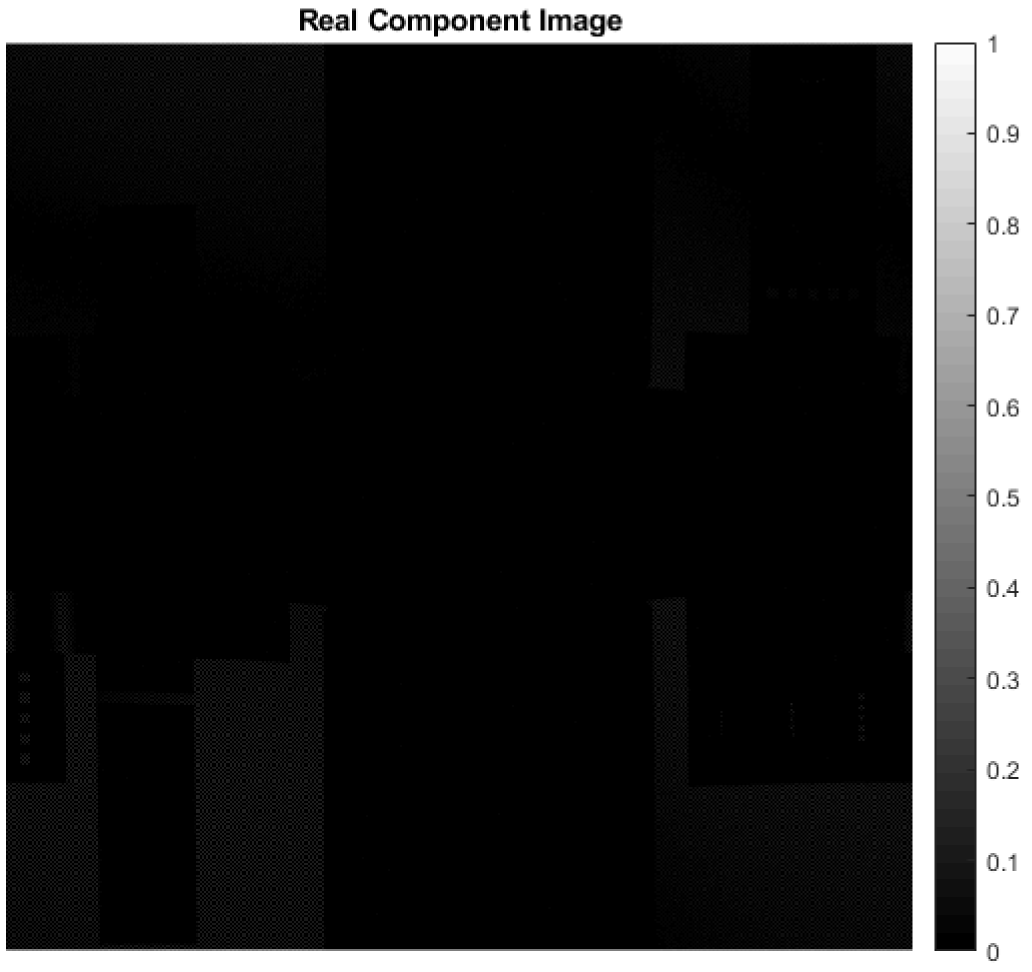
Part b)

Reconstruct the image.

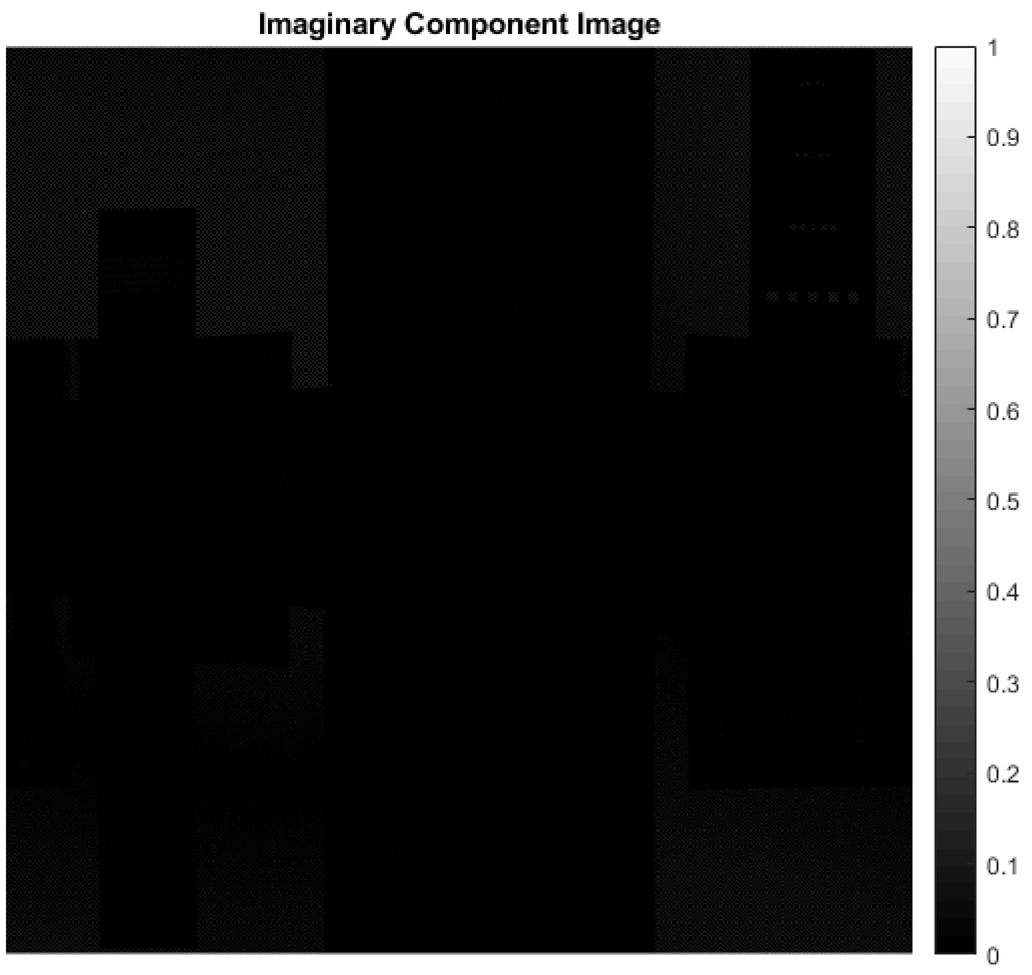
```
%Reconstruct
phant = ifft2(PHANT);
```

Provide the magnitude, phase, real, and imaginary channel of the data in object space.

```
%Show in image space
figure; imshow(real(phant)); title('Real Component Image'); truesize; colorbar
```

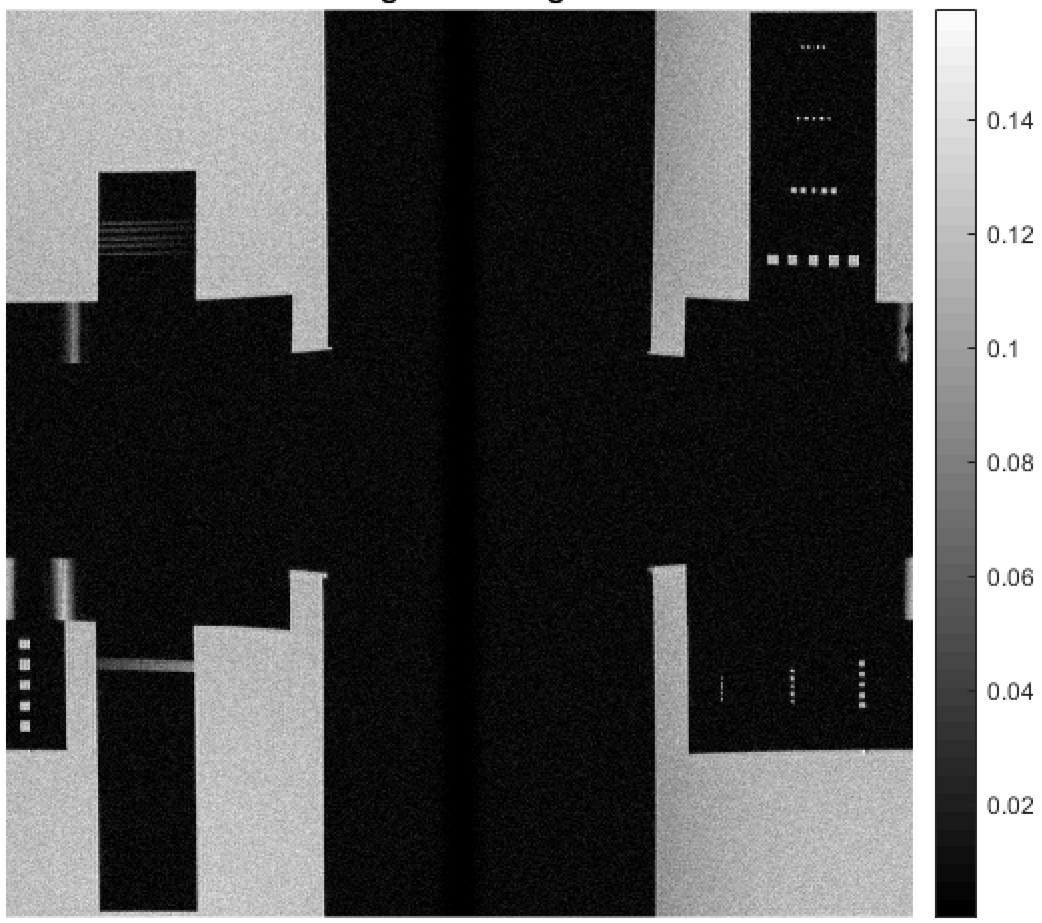


```
figure; imshow(imag(phant)); title('Imaginary Component Image'); truesize; colorbar
```

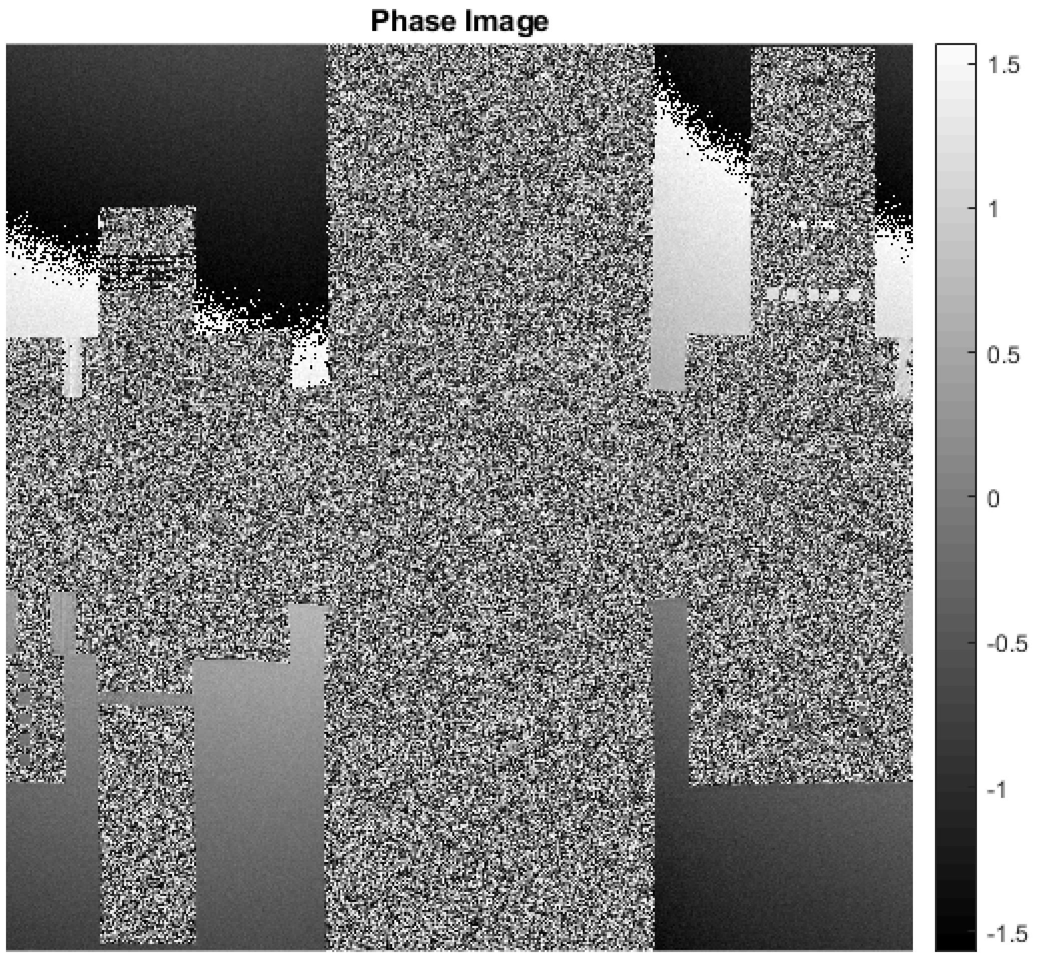


```
figure; imshow(get_mag(phant),[]); title('Magnitude Image'); truesize; colorbar
```

Magnitude Image



```
figure; imshow(get_phase(phant),[]); title('Phase Image'); truesize; colorbar
```



Comment on similarities or differences of these image space data to the ones numerically generated in problem 1.

In terms of similarities, both image 1 (object in problem 1) and image 2 (object in this problem) phantoms contain relatively sharp edges.

In terms of differences, the real component of the signal in image 2 is not the same as the magnitude image as it was in image 1. For image 2, there is a non-zero phase when the image is reconstructed (i.e., the reconstructed image dataset is not entirely real). Additionally, there seems to be much more noise in image 2 compared to image 1. Lastly, the phantom in image 1 has much more high frequency signal than image 2, where most of the image is composed of large rectangles with only a few high frequency line pair tests.

Part c)

Calculate the time required to scan this image.

Excluding the time from the pre-scan, the scan time = TR * # Phase Encodes

```
ScanTime = Npe*TR        % (s)
```

```
ScanTime = 222.7200
```

The scan time is approximately 217 seconds or 3 minutes and 37 seconds.

Part d)

Calculate the peak gradient amplitudes for the readout gradient Gx, the phase encoding gradient Gy, and the slice encoding gradient Gz with the following parameters: rectangular phase encoding gradient with duration of Ty = 1ms and a slice selection gradient with a bandwidth of 1250 Hz.

Note that most of the variables were defined in the beginning of the problem.

```
Ty = 0.001;         % y-gradient duration (s)
BWss = 1250;        % Slice select bandwidth (Hz)
```

X Gradient

We know that our receiver bandwidth is directly related to the ADC sampling rate, which is in turn defined by the maximum frequencies we expect to see in our signal due to the x-gradient.

```
deltaTx = 1/BWr;        % sampling rate
Tx = Nfe*deltaTx;       % x-gradient duration (s)
```

Furthermore, we know that the receiver bandwidth and the FOV in image space are directly related by the maximum gradient strength. Assuming a perfectly square gradient:

$$G_x * \Delta x = \Delta B = \frac{\Delta f}{\gamma}$$

$$G_x * \mathrm{FOV}_x = \frac{\mathrm{BW}_{\mathrm{receiver}}}{\gamma}$$

```
Gx = BWr/(FOVx*gammaBar)        % Maximum gradient strength x-dimension (T/m)
```

```
Gx = 0.0027
```

The maximum x gradient strength is 2.7 mT/m.

Y-Gradient

If we assume $k_{\max}$ are equivalent in both x and y, we can calculate what $k_{\max}$ would be in x, calculate this for y, and back-calculate the gradient strength since we know the gradient duration (Ty, given above). Since the readout gradient runs from -$k_{\max}$ to +$k_{\max}$, $|k_{\max}|$ can be calculate as follows:

$$|k_{\max}| = \gamma\, G_x \frac{T_x}{2} = \gamma\, G_y T_y$$

$$G_y = G_x \left(\frac{T_x}{2T_y}\right)$$

```
Gy = Gx*(Tx/(2*Ty))           % Maximum gradient strength y-dimension
```

```
Gy = 0.0334
```

The maximum y gradient strength is 3.34 mT/m.

Z-Gradient

Again, we know that bandwidth, slice thickness, and gradient strength are related by:

$$G_z * \Delta z = \Delta B = \frac{\Delta f}{\gamma}$$

$$G_z = \frac{\mathrm{BW}_{\mathrm{slice}}}{\Delta z * \gamma}$$

```
Gz = BWss/(deltaZ*gammaBar)      % Maximum gradient strength z-dimension.
```

```
Gz = 0.0147
```

The maximum z gradient strength is 1.47 mT/m.

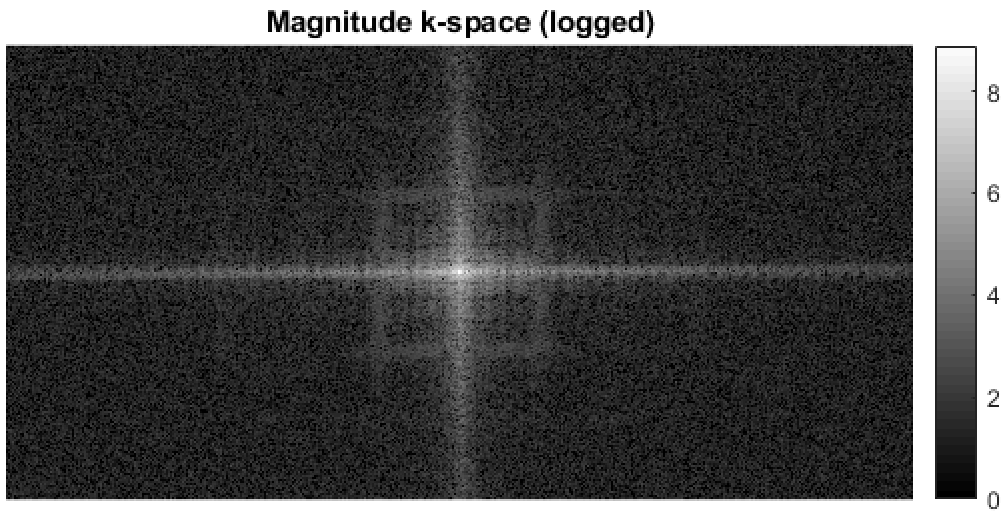
<u>Part e)</u>

Simulate a ‘partial Fourier’ acquisition that samples 50% of the phase encodes by using only the central lines (rows) of k-space – just like you did in problem 1b).

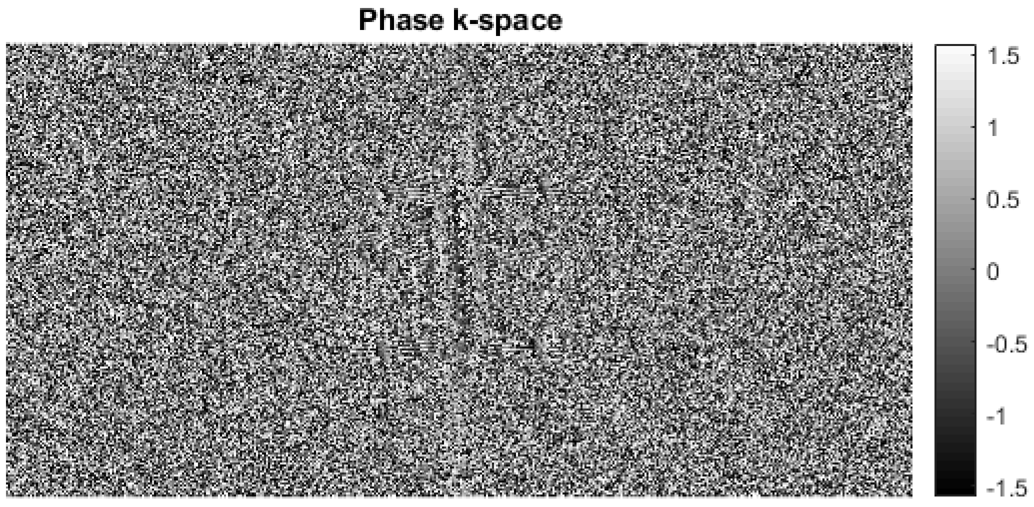
```
[Nx,Ny] = size(PHANT);
index = (Ny*0.5)/2;
%Cut the data in half
cutPHANT = PHANT((index+1:end-index),:);
```

Provide the resulting magnitude and phase image.

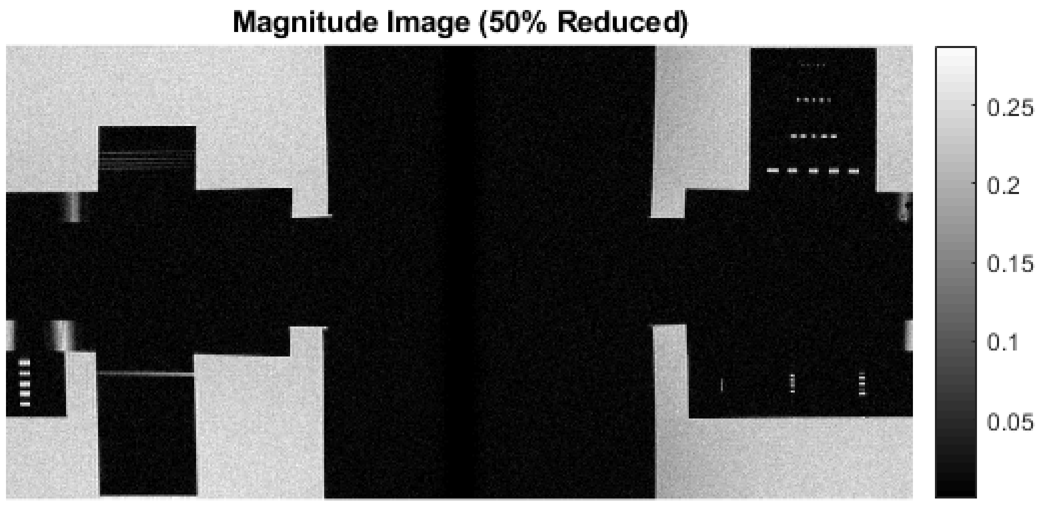
```
MAG2e = get_mag(cutPHANT);
PHASE2e = get_phase(cutPHANT);

figure; imshow(log(MAG2e),[]); title('Magnitude k-space (logged)'); truesize; colorbar
```



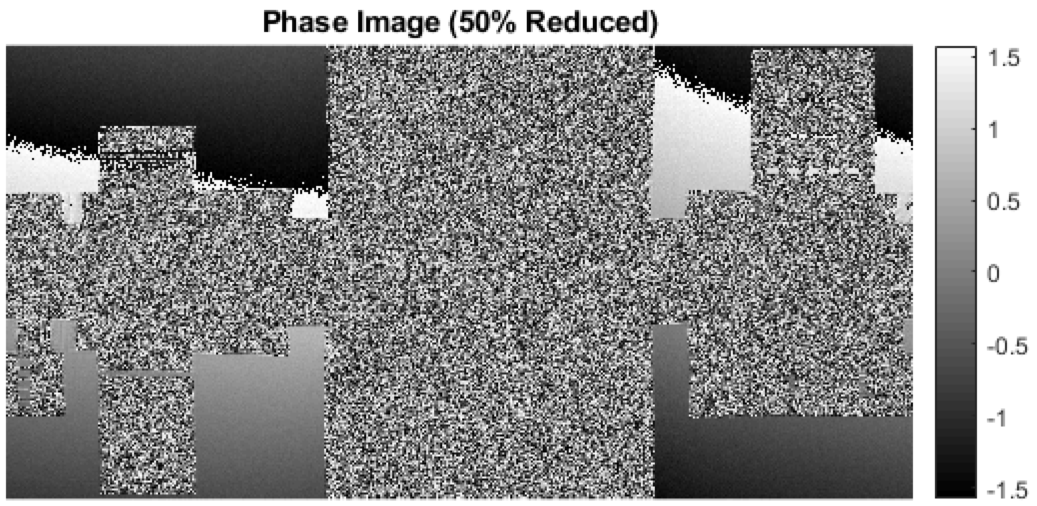
```
figure; imshow(PHASE2e,[]); title('Phase k-space'); truesize; colorbar
```



```
cutphant = ifft2(cutPHANT);
mag2e = get_mag(cutphant);
phase2e = get_phase(cutphant);

figure; imshow(mag2e,[]); title('Magnitude Image (50% Reduced)'); truesize; colorbar
```



```
figure; imshow(phase2e,[]); title('Phase Image (50% Reduced)'); truesize; colorbar
```



Compare the image you generated with all k-space data to this one.

Just as in question 1b, there is decreased spatial resolution in the y-dimension due to the reduction in phase encodes.

Part f)

Provide SNR estimates for both acquisition and discuss your findings.

The full SNR equation is as follows:

$$\mathrm{SNR} = K\left(\frac{\mathrm{FOV}_x}{N_x} * \frac{\mathrm{FOV}_y}{N_y} * \Delta z\right)\sqrt{\frac{N_x * N_y * \mathrm{NEX}}{\mathrm{BW}_{\mathrm{receiver}}}} = (\Delta x \Delta y \Delta z)\left(\sqrt{\left(\frac{(N_x N_y)}{\mathrm{BW}_{\mathrm{receiver}}}\right)}\right)$$

Here, K is a constant that describes hardware dependant factors (coil type, noise power spectrum, field strength-dependent parameters, etc.). In our case, the SNR equation will be *approximated* since we do not know most of the hardware dependant factors.

The equation for SNR from part b will be calculated below.

```
deltaX = FOVx/Nx;
deltaY = FOVy/Ny;
NEX = 1;

SNRb = (deltaX*deltaY*deltaZ)*(sqrt((Nx*Ny)/(BWr)))
```

```
SNRb = 8.7692e-10
```

The equation for SNR from part f will be calculated below. Note the change in Δy and Ny. The other parameters remain the same.

```
Nyy = 0.5*Ny;
deltaYY = FOVy/Nyy;

SNRf = (deltaX*deltaYY*deltaZ)*(sqrt((Nx*Nyy)/(BWr)))
```

```
SNRf = 1.2402e-09
```

To compare, we can simply take the ratio.

```
ratio = SNRf/SNRb
```

```
ratio = 1.4142
```

Thus, there is a $\sqrt{(2)}$ improvement in SNR when k-space is reduce by 50% but there is also a reduction in spatial resolution in the y-dimension.

Problem 3

The data in this series were acquired during an MRI scan of a volunteer. Each data set consists of 9 sagittal slices covering one half of the head of the volunteer.

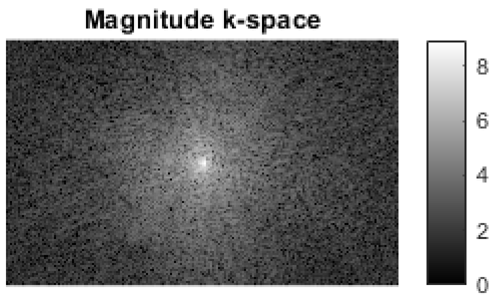
Part a)

Read in slice 3 from the raw data file T1_350.mri, a T1-weighted gradient echo sequence with the following parameters: TR = 350 ms, TE = 6.8 ms, flip angle = 60 deg., receiver bandwidth = 15.63 kHz, field of view: 24 cm x 24 cm, # of samples in readout = 256, # of phase encodes = 160, slice thicknes = 5mm.

```
HEAD = read_raw('T1_350.mri',256,160,3);
```

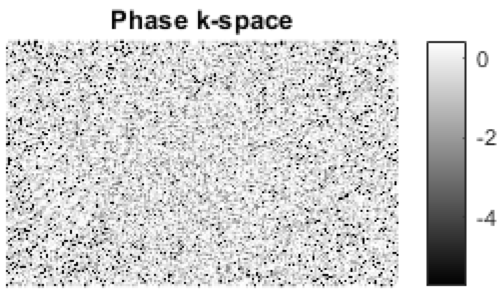
Provide an image of the k-space magnitude in a logarithmic scale.

```
figure; imshow(log(get_mag(HEAD)),[]); title('Magnitude k-space'); truesize; colorbar
```



```
figure; imshow(log(get_phase(HEAD)),[]); title('Phase k-space'); truesize; colorbar
```

```
Warning: Displaying real part of complex input.
```



Compare these raw data with the raw data from the phantom scan e.g. in 2a).

There is no high intensity "square" shape surrounding the center of k-space. Instead, the intensity seems to uniformly decrease as we move further from k-space. Like the other k-space datasets, there is a high amount of signal in the center of k-space, which is expected.

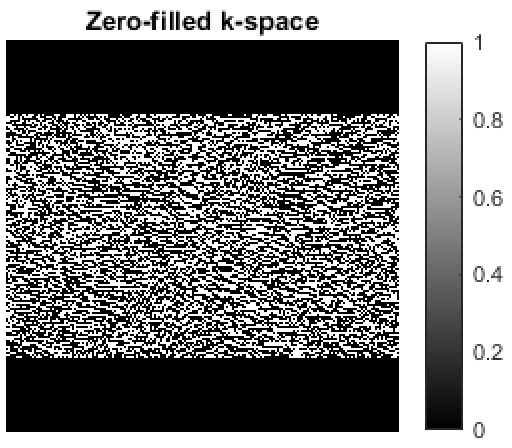
Part b)

Reconstruct the image with a proper aspect ratio as you learned in 1c). Provide an image of the magnitude in object space.

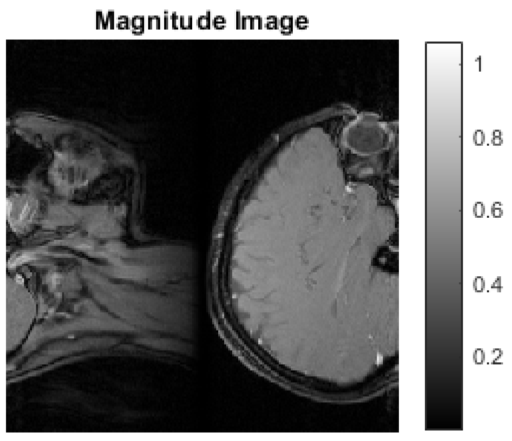
```
[Ny,Nx] = size(HEAD);
```

```
rows = 0.5*(Nx-Ny);
extra = zeros(rows,Nx);
HEAD2 = [extra; HEAD; extra];
figure; imshow(HEAD2); title('Zero-filled k-space'); truesize; colorbar
```

```
Warning: Displaying real part of complex input.
```



```
head2 = ifft2(HEAD2);
figure; imshow(get_mag(head2),[]); title('Magnitude Image'); truesize, colorbar
```



Part c)

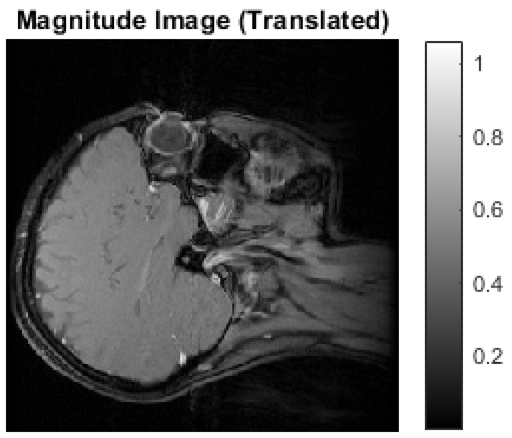
Most likely you will see the image in2.5b not properly centered in the FOV. This artifact is caused because the scanner software inverts the signal of every other phase encode acquisition (multiplied every other horizontal line of k-space by –1). The reason for this was the sensitivity of early scanners to sampling errors for the DC k-space sample.

Now, multiply every other column in the raw data set by –1 and perform an inverse two dimensional fft without using the fftshift algorithm in your reconstruction.

```
HEAD3 = HEAD2;
for i = 1:2:256
    HEAD3(:,i) = HEAD3(:,i)*-1;
end

head3 = ifft2(HEAD3);
```

View the magnitude of the reconstructed image.

```
figure; imshow(get_mag(head3),[]); title('Magnitude Image (Translated)'); truesize; colorbar
```



Can you explain why multiplying by a –1, 1 –1, 1 …. across the lines and columns of k-space properly positions the data? Hint: -1, 1, -1, 1 ….is equal to exp(-iπn).

This works becuse $e^{(-i\pi n)}$ is essentially a rotation (phase shift) in k-space. This equates to a translation in the image domain, shifting our head to the correct orientation.