# Lab Report

# ENSC 477 - Biomedical Image Acquisition

Lab 4

Group 9


| Group Member | Student ID |
| --- | --- |
| Benjamin Martin | 301347720 |
| Daria Zhevachevska | 301326410 |

School of Engineering

Faculty of Applied Sciences

Simon Fraser University


Fall 2020

# Introduction

Magnetic resonance imaging is a technique that utilizes the naturally occurring magnetic moments created by the nuclear spin of atoms with odd numbered protons in their nucleus. Through an elegant exchange of electromagnetic manipulation, the proton density, transverse relaxion and longitudinal relaxation time are used to generate signals that produce a volume image which contrasts the different tissue types in the target.

Alignment of the microscopic nuclear spin systems is done by the main bore of the MRI system. Before the external magnetic field $B_0$ is applied, the magnetic moments have no preference in the orientation they spin. When the main bore, a super-conducting magnet constructed as a solenoid, is turned on the macroscopic magnetic field changes to and extraordinarily strong uniform magnetic field in the z direction. $B_0$ is an extremely strong field, ranging from 0.5 T to 9 T field strengths (1.5 T is typical for medical imaging), which torques all microscopic nuclear spin systems in the solenoid to align around the z-axis and precess at the Larmor frequency.

To select only a portion of the sample along the z-axis, a gradient magnetic field along the z-axis is introduced by gradient coils which result in a slight variation in precession frequency of the magnetic moments. A magnetic pulse, $B_1$, which consists of the same frequencies as the precession frequencies of the sample along a portion of the z-axis, can then be applied to tip that portion of magnetic moments into the transverse xy-plane. This RF-excitement resonates with the precession frequency of the particular slice because $B_1$ will push towards the xy-plane whenever the magnetic moment is oriented with either the positive or negative y-axis. This intricate process, called forced precession, traces a spiral path with the magnetic moments along a sphere and is calibrated to have the required amplitude and duration to tip the moments 90° transversely.

With a range of Larmor frequencies present in a slice due to the gradient magnetic field along the z-axis, $G_z$, the spins of the magnetic moments will be out of phase after the process of tipping which would result in a weak net magnetic moment. The RF pulse strongest at the centre, and the significant phase accumulation occurs across a period of half the length of the RF pulse. The simplest solution is a refocusing lobe of negative amplitude to $G_z$ and half the duration of the RF pulse applied to rephase all spins in the slice. Any refocusing lobe signal shape with an integral value equal to the negative of half the integral value of the RF slice selection pulse will accomplish rephasing though, so a faster refocusing lobe is desirable.

With the slice of the z-axis selected, sampling is done in the frequency domain where a magnetic field gradient is produced by two more gradient coils. The y-axis gradient encodes the phase of the precessing atoms and the gradient along the x-axis encodes the frequency of precessing atoms. Unlike the, z-gradient, however, these gradients must be applied for only a short while and then turned off (although the x-axis gradient occurs simultaneously to the readout).

Radio frequency coils detect signals as induced currents produced by the projection of large net magnetic moments on the xy-plane. When the microscopic magnetic moments are in phase an analog to digital converter samples the signals as representative pixels in k-space which can be 2-D inverse Fourier transformed into an image in the spatial domain.

# Part 1: Processing MRI data

## Method

Data in k-space of a mystery sample was provided by Andrew Yung at UBC's research centre from a 7 T MRI. The sample consists of 27 axial slices, each with 256 x 256 samples in the uv-plane. Partially reconstructed, and fully reconstructed, images are produced from both 1D and 2D inverse Fourier transforms and the results are compared.

Given the resolution of the reconstructed image is 0.117 x 0.117 mm in-plane with 0.5 mm slice thickness, and knowing pixels are preserved in the 2D Fourier transform, the k-space axes are calculated:

$$0.117mm = FOV_x = \frac{1}{\Delta u}, \qquad \Delta u = 8.547\ mm^{-1},$$

$$U = \Delta u \times N_a = 8.547\ mm^{-1} \times 256 = 2188\ mm^{-1}$$

$$0.117mm = FOV_y = \frac{1}{\Delta v}, \qquad \Delta v = 8.547\ mm^{-1}$$

$$V = \Delta v \times N_a = 8.547\ mm^{-1} \times 256 = 2188\ mm^{-1}$$

After processing the volume image, a 2D image of an axial slice is present by holding the z-coordinate constant. Then a coronal image is produced by holding y constant, and a sagittal image is produces by holding x constant.

Since the multiplication in the Fourier space is same as convolution in the spatial domain, the filtering was done in the Fourier space. The filters were implemented in a simplest way by creating a 256X256 matrix with a filled circle in the centre where radius = cut-off frequency. The examples of the filters are shown below. Each axial slice of the original data was multiplied with a filter matrix, the frequencies multiplied with zero were removed. The last step was to inverse transform the filtered volume.

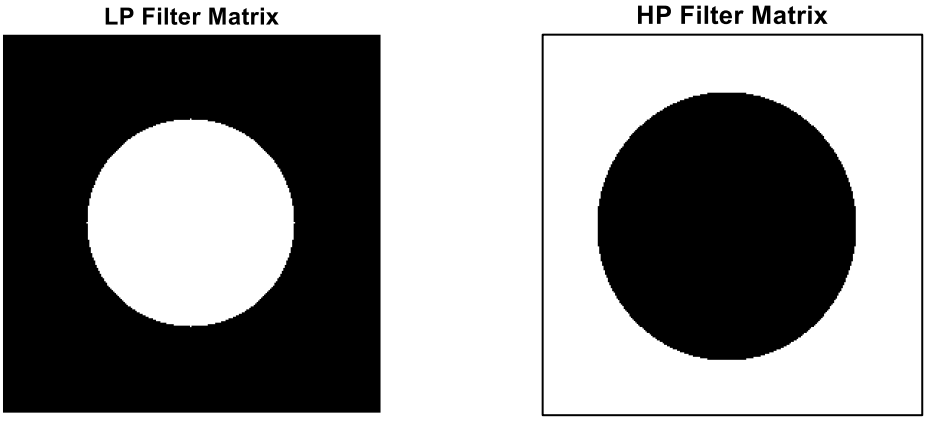


***Figure 1:*** *Low (on the left) and High Pass (on the right) Filter Matrices in the Frequency Domain*

# Results

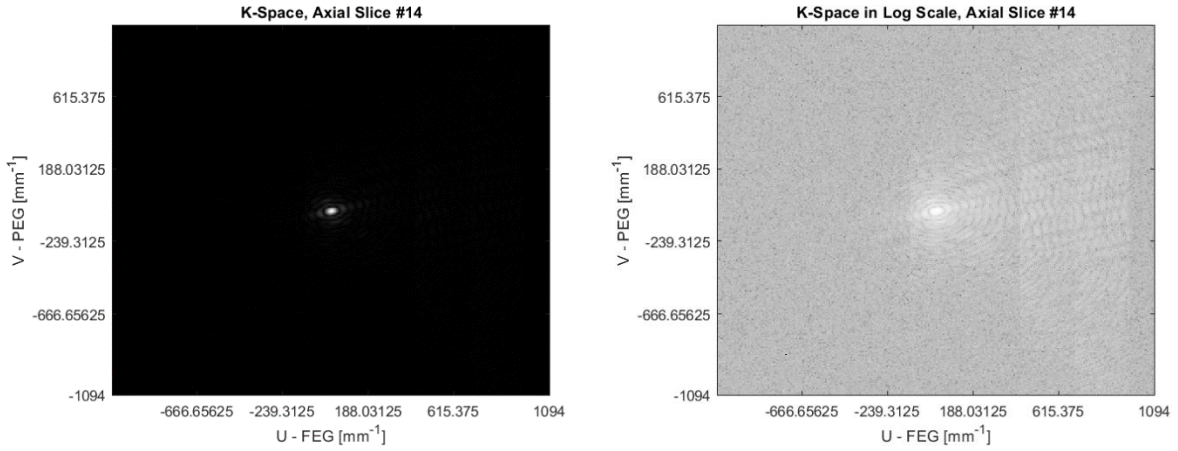


***Figure 2:*** *Linear Scale Fourier Space Data from an Axial Slice #14 (on the left) and Log Scale of the Same Slice (on the right)*

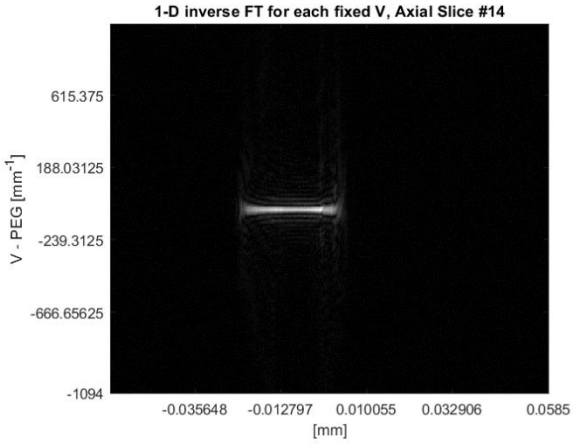


***Figure 3:*** *An inverse 1D Fourier Transform with Fixed V to Produce a Partially Reconstructed Image of Axial Slice #14*

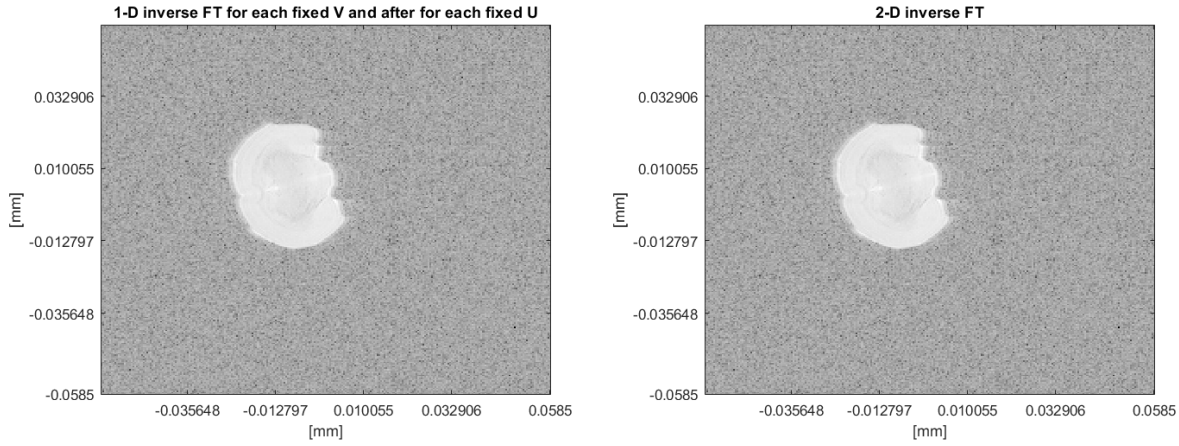


***Figure 4:*** *A Second Inverse 1D Fourier Transform for Fixed U Produced a Completed Reconstruction of Axial Slice #14 (on the left). A 2D Inverse Fourier Transform Produced a Reconstructed Axial Slice #14 (on the right)*

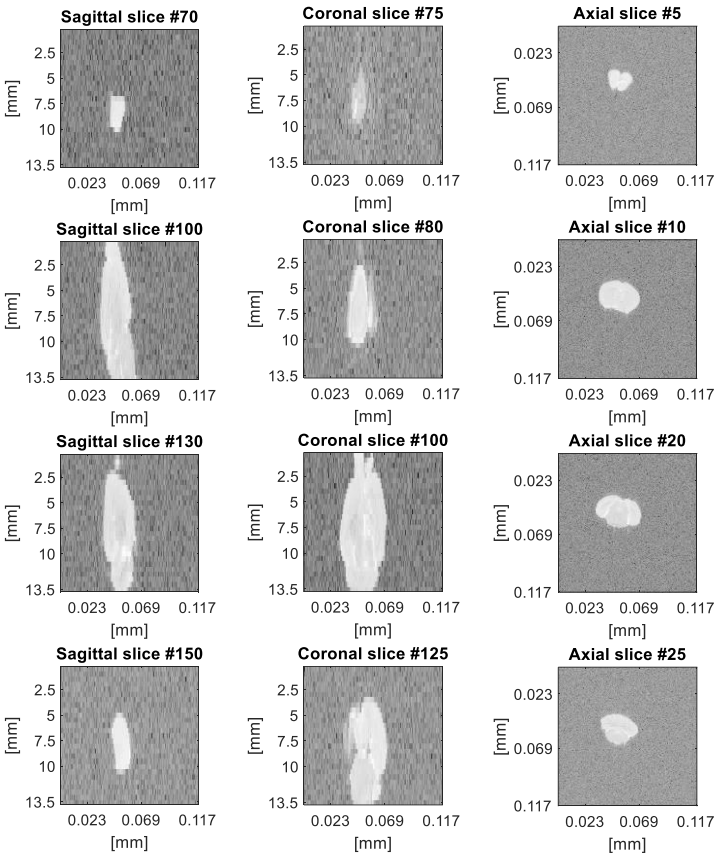


***Figure 5:*** *Sagittal Slices (on the left), Coronal Slices (in the middle) and Axial Slices (on the right) of the Reconstructed Object*

After going through the reconstructed volume from each of the three planes, the object resembles a brain, in addition, because the resolution of the brain is less than 0.117x0.117x13.5 mm, it was concluded that this is a brain of a very small animal. When looking down on the xy-plane, we see the cross-section of the spinal disks. Unfortunately, when looking at the sagittal and coronal slices it is not clear what the object is. In order to solve this problem, we would need to increase the number of axial sample points.

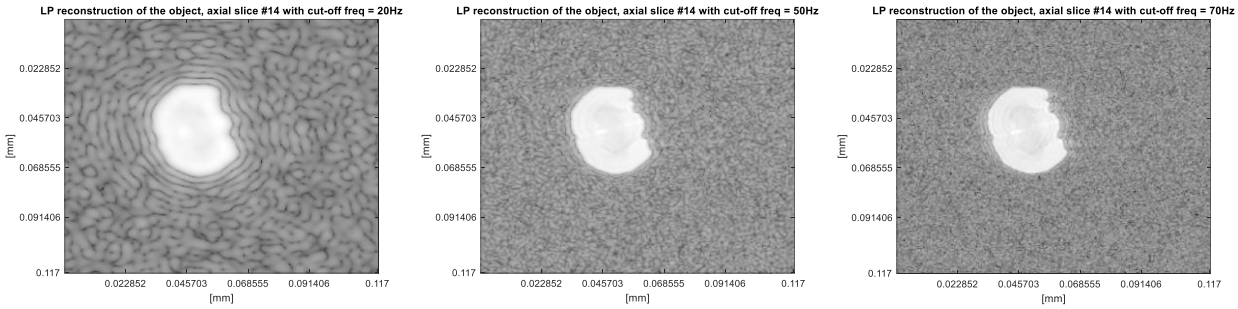


***Figure 6:*** *Low Pass Filtering of the Reconstructed Axial Slice #14 with Different Cut-Off Frequencies*

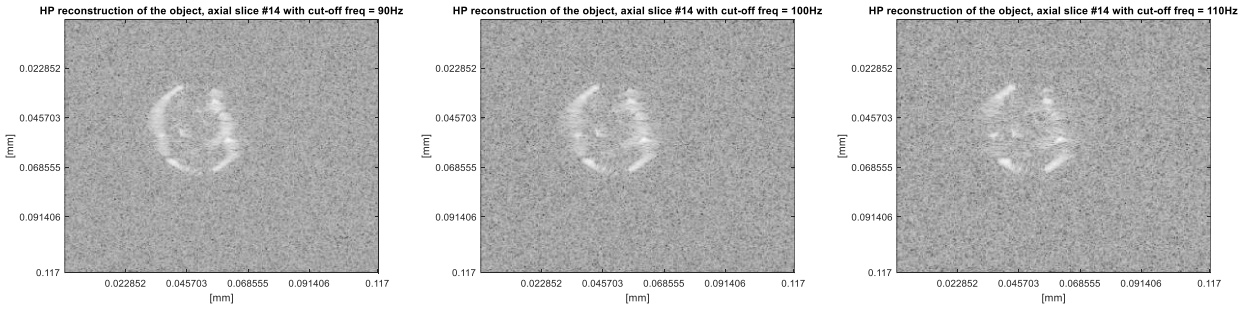


***Figure 7:*** *High Pass Filtering of the Reconstructed Axial Slice #14 with Different Cut-Off Frequencies*

## Discussion

The logarithmic scale is used to compress the dynamic range of the Fourier spectrum which has values [0, 10^6] or even higher. When displaying the image in Fourier space, the high intensity values will dominate, this will cause the loss of visual details, the pixels with darker intensity. After the logarithmic compression, the darker pixels become brighter and the brighter pixels become less bright, thus making the details present in darker or gray areas of the image more visible to human eyes.

The results in Figure 4 show that the complete reconstruction of the object using two inverse 1D Fourier transform along the two dimensions is exactly the same as the reconstruction of the using the inverse 2D Fourier transform. The 2-dimensional inverse transform is computed by first transforming each row/column, replacing each row/column with its spatial values and then inverse transforming each column/row, replacing each column/row with its spatial values. Therefore, a 2D inverse transform of NxN image involves 2N 1D inverse transforms. The one difference between these two methods of object reconstruction is the computation time. An inverse FFT is much faster than two 1D inverse FTs.

$$f(x,y) = \frac{1}{NN}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} F(u,v)e^{j2\pi(u\frac{x}{N}+v\frac{y}{N})} = \frac{1}{NN}\sum_{x=0}^{N-1}[\sum_{y=0}^{N-1} F(u,v)e^{\frac{j2\pi vy}{N}}]\, e^{\frac{j2\pi ux}{N}}$$

Figure 7 and 8 display the results applying low and high pass filters on the image. Low pass reduces the edged content and the image will look smoothened out. On the other side, the high pass has the opposite effects. Removing the low frequencies will leave an image only with sharp intensity changes which are the fine-scale details and noise in the spatial domain image, the outlines and most of the background of the object is lost. By manipulating with the value of the cuff-off frequency, we decide what range of frequencies is to be blocked. When selecting 70Hz as the cut-off frequency the obtained images seem to have less noise in the background and still retain the outline and details of the brain.

If one of the pixels in k-space was corrupted and spuriously recorded with an amplitude of 10^10, once inverse Fourier transformed that k-space pixel's frequency and phase would superimpose during image reconstruction with too much intensity. What this would look like depends on whether the image was

scaled logarithmically or not. If not, the intensity would be so great all the noticeable contrast would the stripe pattern from the corrupted k-space pixel. If logarithmically scaled, the intensity would be less than 10 times greater than the intensities of the other stripe patterns. Recalling there are 256 x 256 pixels, this effect would be negligible after logarithmic scaling.

# Part 2: Processing MRI data

## Method

A set of 3 human brain MRI data is download from OASIS database. An axial, coronal, sagittal slices and a volumetric body mock-up are generated and displayed using FIJI, a 3D visualization software.

One of the lateral verticles is used to roughly estimate its major and minor axis. An ellipse around the ventricle is imagined and 2 lines through the center are drawn, where the major axis is the longer line and the minor axis the shorter one. To estimate the length of each axis, the number of pixels along thee axis is calculated and multiplied by the voxel resolution, which is obtained from the .txt data files and is 1.0 x 1.0 x 1.25 mm for all 3 volumes.

## Results

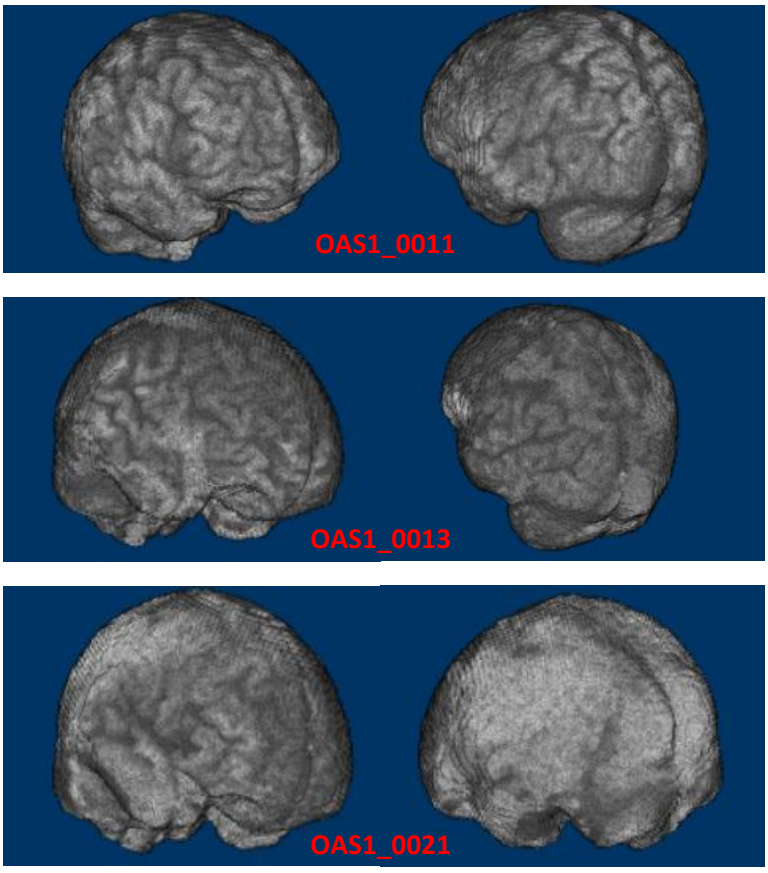


***Figure 8:*** *The Volumetric View of 3 Subjects*

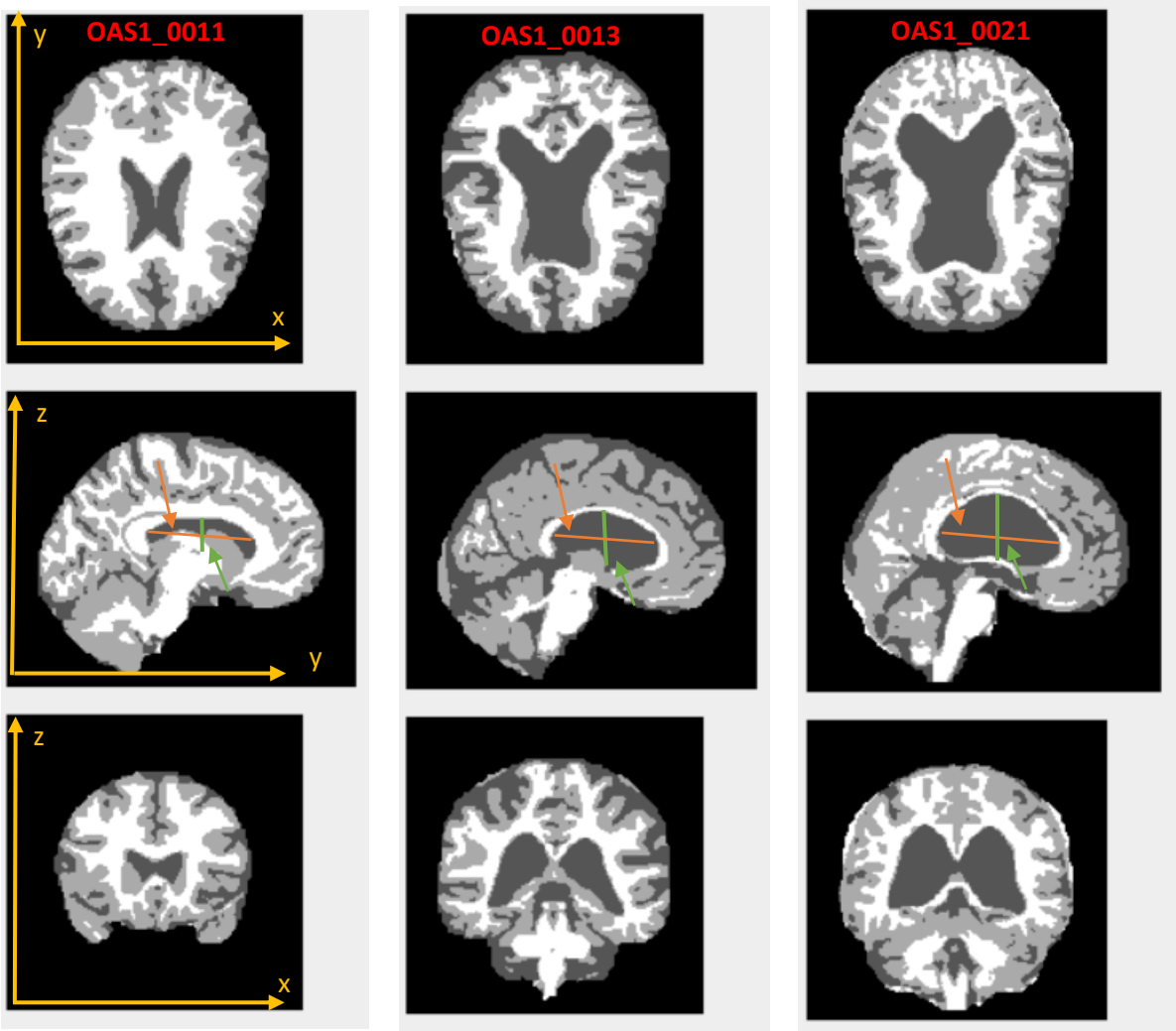


***Figure 9:*** *The Axial (top row), Sagittal (middle row) and Coronal (bottom row) Views of 3 Subjects. The Orange Line is the Estimate of the Major Axis and the Green Line is the Estimate of the Minor Axis of the Brain Ventricle*

| OASIS ID | Age | MMSE | Min Axis [mm] | Max Axis [mm] |
|---|---|---|---|---|
| OAS1_0011 | 52 | 30 | 17.5 | 60 |
| OAS1_0013 | 81 | 30 | 38.75 | 62 |
| OAS1_0021 | 80 | 23 | 45 | 67 |

***Table 1:*** *Summary about the MRI Data Cohort*

## Discussion

Three dimensional T1-weighted MRI were acquired from a young and cognitively normal brain, an elderly and cognitively-normal brain, and an age-matched cognitively-impaired brain. The images show that in general older adults have larger ventricles than the young adults. The major axes were measured to have around the same lengths 60-67mm, although, the minor axes are significantly different. The 50-year-old patient has the smallest ventricle, with a minor axis of only 17.5mm. In contrast the older subjects were measured to have twice longer minor axes. When comparing an elderly and cognitively-normal brain and an age-matched cognitively-impaired brain it is obvious that the ventricles of the later are more expanded along the minor axis.

A common practice for assessing cognitive impairments is the use of Mini-Mental State Examination (MMSE), which is a 30-point questionnaire. The biggest advantage of this test is the ease of use, a very short completion time and reliability[1]. Although, it fails to detect mild cognitive impairment and cannot adequately discriminate patients with mild Alzheimer's disease from normal patients[2].

MRI scans allow clinicians to have a very detailed view of nearly every part of the body in slices. MRI is extensively used to image brain to diagnose strokes, tumors, brain injuries and potentially cognitive abnormalities, since the topographic distribution of regions in the brain that are responsible for cognition shows the atrophy or abnormalities of the brain tissue.

## Conclusion

MRI performs object sampling in the k-space which can be 2D inverse Fourier transformed into an image in the spatial domain. The separability property of the 2D inverse FT makes it possible to compute the inverse by taking the 1D row FT followed by a 1D column FT or by a 2D Fast Fourier Transform, which is a computationally faster way.

Because the MRI data is large, it is most time and memory efficient to first perform filtering in the k-space and then inverse transform. The spatial convolution is done by point-wise multiplication of the filter matrix with k-space acquired data.

When the object is reconstructed, the human body is visualized in thin and adjacent slices. There are 3 types of views: from the side is the sagittal view, from the front is the coronal and from top down is the transverse view. It is important to see all three views since they all carry different structural and morphological information.

---

[1] https://journals.lww.com/alzheimerjournal/Abstract/2000/07000/The_Severe_Mini_Mental_State_Examination__A_New.8.aspx

[2] https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1532-5415.1992.tb01992.x

SFU FACULTY OF SCIENCE

## Data

http://www.oasis-brains.org
OASIS: Cross-Sectional: https://doi.org/10.1162/jocn.2007.19.9.1498

## Appendix

```
%%%%%%%%%%%%%%%%%%% Part 1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Load Data
addpath('C:\Users\Dasha\Desktop\Fall 2020\ENSC
477\Labs\Lab4-MRI\DataFiles\');
kSpaceData = load('sample_kspace.mat');
kSpaceArray = kSpaceData.kspace_input;

%% Show an image of the K-space of one axial slice
kSpaceA1 =  kSpaceArray(:,:,14);

%change the x and y axis labling
val0 = [50 100 150 200 256]/256 * 2188;
charVal0 = strsplit(num2str(val0)); %create char values for
axis
figure(1)
imagesc(abs(kSpaceA1)); colormap('gray');
xticks([50 100 150 200 256]); yticks([50 100 150 200 256]);
set(gca,'XTickLabel', charVal0); set(gca,'YTickLabel',
charVal0);
xlabel('U - FEG [mm^-^1]');
ylabel('V - PEG [mm^-^1]');
title('K-Space, Axial Slice #14');
figure(2)
imagesc(log(1+abs(kSpaceA1))); colormap('gray');  % display
log scale
xticks([50 100 150 200 256]); yticks([50 100 150 200 256]);
set(gca,'XTickLabel', charVal0); set(gca,'YTickLabel',
charVal0);
xlabel('U - FEG [mm^-^1]');
ylabel('V - PEG [mm^-^1]');
title('K-Space in Log Scale, Axial Slice #14');

%% Take 1-D inverse FT for each fixed V
UFTA1 = ifft(kSpaceA1, [], 2);
figure(3);
imagesc(abs(UFTA1)); colormap('gray');
```

```
%change the x and y axis labling
val = [50 100 150 200 256]/256 * 0.117;
charVal = strsplit(num2str(val)); %create char values for
axis
set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charVal0);
xticks([50 100 150 200 256]); yticks([50 100 150 200 256]);
xlabel('[mm]'); ylabel('V - PEG [mm^-^1]');
title('1-D inverse FT for each fixed V, Axial Slice #14');

%% Take 1-D inverse FT for each fixed U after
FTA1 = ifft(UFTA1, [], 1);
FTA1log = log(1+abs(FTA1));
figure(4);
colormap('gray');
imagesc(FTA1log);
title('1-D inverse FT for each fixed V and after for each
fixed U');

%change the x and y axis labling
xticks([50 100 150 200 256]); yticks([50 100 150 200 256]);
set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charVal);
ylabel('[mm]'); xlabel('[mm]');

%% Take 2-D inverse FT and compare
FT2dA1 = ifft2(kSpaceA1);
FT2dA1log = log(1+abs(FT2dA1));
figure(5);
colormap('gray');
imagesc(FT2dA1log);
title('2-D inverse FT');

%change the x and y axis labling
xticks([50 100 150 200 256]); yticks([50 100 150 200 256]);
set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charVal);
ylabel('[mm]'); xlabel('[mm]');


%% 2-D inverse FT all axial slices
volumeImage = zeros(size(kSpaceArray));
for i = 1:size(kSpaceArray,3)
    volumeImage(:,:,i) = ifft2(kSpaceArray(:,:,i));
```

```
end


%display along the x-axis
val = [5 10 15 20 27]/27 * 13.5;  %create char values for
z-axis
charValZ = strsplit(num2str(val));

figure(6)
colormap('gray');
imagesc(log(1+abs(squeeze(volumeImage(128,:,:))')));
title('Reconstruction of the object, sagittal slice #128');

%change the x and y axis labling
xticks([50 100 150 200 256]);  yticks([5 10 15 20 27]);
set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charValZ);
xlabel('[mm]'); ylabel('[mm]');

% for i = 50:160
%     imagesc(log(1+abs(squeeze(volumeImage(i,:,:))')));
%     title(['Reconstruction of the object, sagittal slice
#', num2str(i)]);
%     xticks([50 100 150 200 256]);  yticks([5 10 15 20
27]);
%     set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charValZ);
%     xlabel('[mm]'); ylabel('[mm]');
%     pause(0.3);
% end

%display along the y-axis
figure(7)
colormap('gray');
imagesc(log(1+abs(squeeze(volumeImage(:,128,:))')));
title('Reconstruction of the object, coronal slice #128');
xticks([50 100 150 200 256]);  yticks([5 10 15 20 27]);
set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charValZ);
xlabel('[mm]'); ylabel('[mm]');

% for i = 50:150
%     imagesc(log(1+abs(squeeze(volumeImage(:,i,:))')));
```

```
%     title(['Reconstruction of the object, coronal slice
#', num2str(i)]);
%     xticks([50 100 150 200 256]);  yticks([5 10 15 20
27]);
%     set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charValZ);
%     xlabel('[mm]'); ylabel('[mm]');
%     pause(0.3);
% end

%display along the z-axis
figure(8)
colormap('gray');
imagesc(log(1+abs(squeeze(volumeImage(:,:,10)))));
title('Reconstruction of the object, axial slice # 10');
xticks([50 100 150 200 256]); yticks([50 100 150 200 256]);
set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charVal);
ylabel('[mm]'); xlabel('[mm]');

% for i = 1:27
%     imagesc(log(1+abs(squeeze(volumeImage(:,:,i)))));
%     title(['Reconstruction of the object, axial slice #',
num2str(i)]);
%     xticks([50 100 150 200 256]); yticks([50 100 150 200
256]);
%     set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charVal);
%     ylabel('[mm]'); xlabel('[mm]');
%     pause(0.3);
% end

%% LP Filter
% Initialize LP Filter matrix
[xGrid,yGrid] = meshgrid(1:256,1:256);
cutoff = 70;
lpFilter = sqrt((xGrid - 128).^2 + (yGrid - 128).^2) <=
cutoff;
% imshow(lpFilter); title('LP Filter Matrix')

ftrdM1 = kSpaceArray .* lpFilter;

volumeImageLP = zeros(size(ftrdM1));
for i = 1:27
```

```
    volumeImageLP(:,:,i) = ifft2(ftrdM1(:,:,i));
end

%display the LP reconstruction of the object
figure(9)
colormap('gray');
imagesc(log(1+abs(squeeze(volumeImageLP(:,:,14)))));
title(['LP reconstruction of the object, axial slice #14
with cut-off freq = ', num2str(cutoff),'Hz']);
xticks([50 100 150 200 256]); yticks([50 100 150 200 256]);
set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charVal);
ylabel('[mm]'); xlabel('[mm]');
% for i = 1:27
%     imagesc(log(1+abs(squeeze(volumeImageLP(:,:,i)))));
%     title(['LP reconstruction of the object, axial slice
#', num2str(i)]);
%     xticks([50 100 150 200 256]); yticks([50 100 150 200
256]);
%     set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charVal);
%     ylabel('[mm]'); xlabel('[mm]');
%     pause(0.3);
% end

%% HP Filter
% Initialize HP Filter matrix
cutoff = 70;
hpFilter = sqrt((xGrid - 128).^2 + (yGrid - 128).^2) >=
cutoff;
% imshow(hpFilter); title('HP Filter Matrix')

ftrdM2 = kSpaceArray .* hpFilter;

volumeImageHP = zeros(size(ftrdM2));
for i = 1:27
    volumeImageHP(:,:,i) = ifft2(ftrdM2(:,:,i));
end

%display the HP reconstruction of the object
figure(10)
colormap('gray');
imagesc(log(1+abs(squeeze(volumeImageHP(:,:,14)))));
```

```
title(['HP reconstruction of the object, axial slice #14
with cut-off freq = ', num2str(cutoff),'Hz']);
xticks([50 100 150 200 256]); yticks([50 100 150 200 256]);
set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charVal);
ylabel('[mm]'); xlabel('[mm]');
% for i = 1:27
%     imagesc(log(1+abs(squeeze(volumeImageHP(:,:,i)))));
%     title(['HP reconstruction of the object, axial slice
#', num2str(i)]);
%     xticks([50 100 150 200 256]); yticks([50 100 150 200
256]);
%     set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charVal);
%     ylabel('[mm]'); xlabel('[mm]');
%     pause(0.3);
% end


%% Combine LP & HP Filters -> Band Stop Filter
ftrdM3 = kSpaceArray .* lpFilter + kSpaceArray .* hpFilter;
%  figure(15);imshow(ftrdM3); title('BS Filter Matrix')
% figure(11); colormap('gray');
% for i = 1:27
%     imagesc(log(abs(squeeze(ftrdM3(:,:,i)))));
%     pause(0.3);
% end

volumeImageBS = zeros(size(ftrdM3));
for i = 1:27
    volumeImageBS(:,:,i) = ifft2(ftrdM3(:,:,i));
end

%display the BS reconstruction of the object
figure(11)
colormap('gray');
for i = 1:27
    imagesc(log(1+abs(squeeze(volumeImageBS(:,:,i)))));
    title(['BS reconstruction of the object, axial slice
#', num2str(i)]);
    xticks([50 100 150 200 256]); yticks([50 100 150 200
256]);
    set(gca,'XTickLabel', charVal); set(gca,'YTickLabel',
charVal);
```

```
    ylabel('[mm]'); xlabel('[mm]');
    pause(0.3);
end
```