

MRI HW2-FLASH

Presenter :

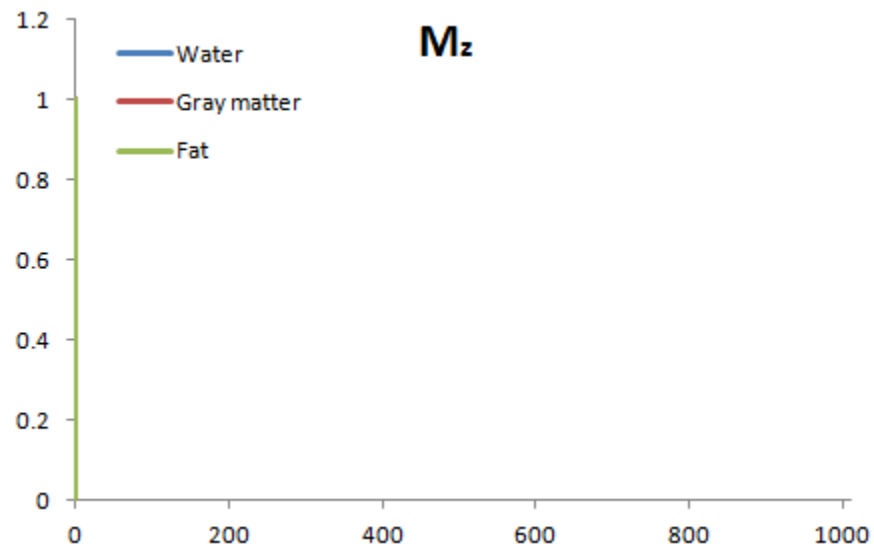
M11107309 何柏昇

OUTLINE

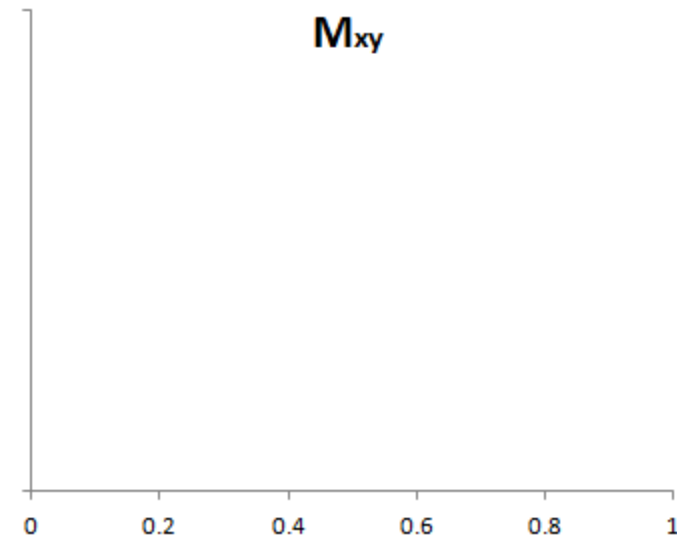
- EX2.1 Simulate The Timing of M_{xy} and M_z
 - Introduction
 - Method
 - Experimental Results
- EX2.2 The Maximum MRI Image Brightness
 - Introduction
 - Method
 - Experimental Results
- EX2.3 Subcortical Structures
 - Introduction
 - Method
 - Experimental Results

Introduction

$$M_z = M * (1 - e^{-t/T1})$$



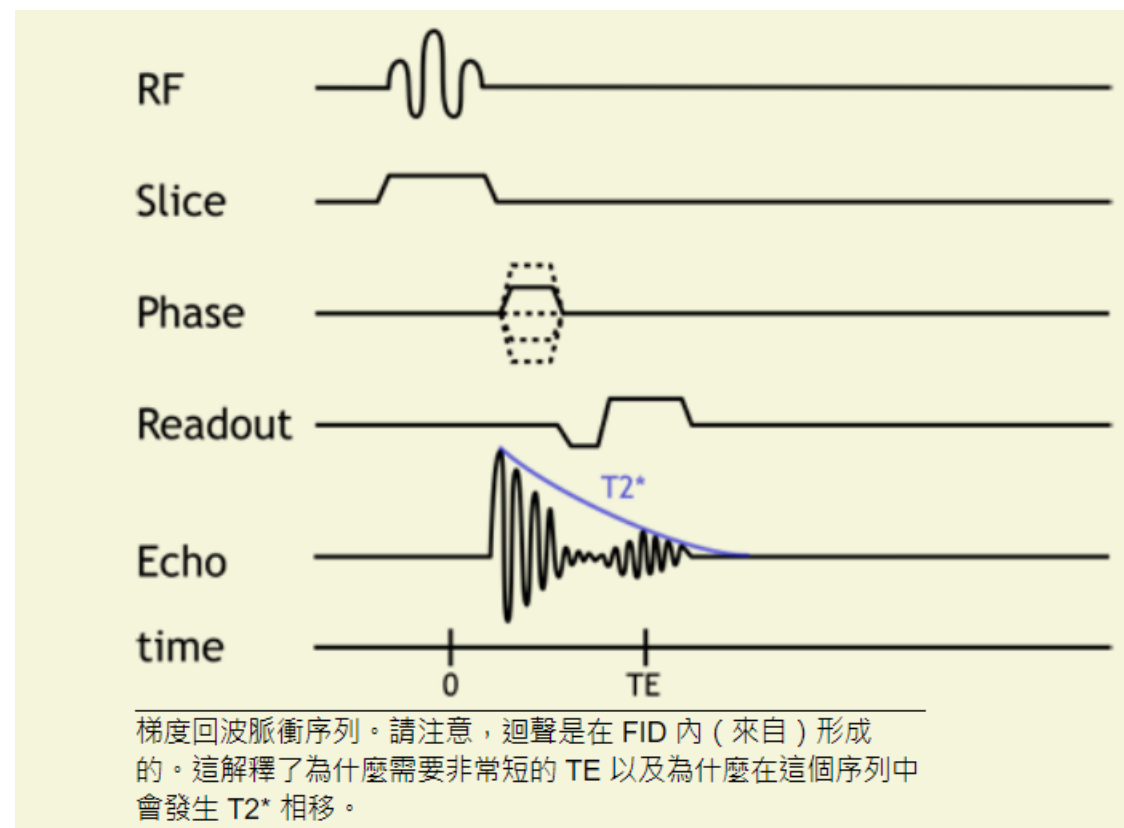
$$M_{xy} = M_0 * e^{-t/T2}$$



Introduction

Gradient Echo

- 由於 T2 效應未被抵消，GRE 序列的信號損失更快，因此通常使用更短的 TE。
- 更短的 TE 允許使用更短的 TR，這增加了 T1 權重



Introduction

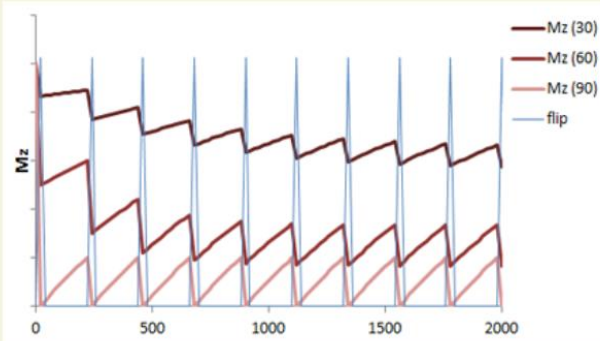


Illustration of the recovery of longitudinal magnetization in a GRE sequence with short TR and varying flip angles over successive applications of the pulse sequence. Given the short TR, there is not much time for the longitudinal magnetization (M_z , red) to recover. Using small flip angles (e.g. 30 degrees, 60 degrees) allows a larger fraction of the longitudinal magnetization to remain, so recovery is shorter. The 90 degree flip angle gives the lowest amount of signal (light red).

$$M_z = \text{終} + (\text{初} - \text{終})e^{-t/T_1}$$

$$\text{初} = \left(\frac{\left(1 - e^{-\frac{TR}{T_1}}\right) \sin \alpha}{1 - e^{-\frac{TR}{T_1}} \cos \alpha} \right) * e^{-TE/T_2^*}$$

事實上信號強度的公式是

- 信號正比於

$$\frac{(1 - e^{-TR/T_1}) \sin \alpha}{1 - e^{-TR/T_1} \cos \alpha} e^{-TE/T_2^*}$$

- α : 偏折角

Method

Initialize

```
% Initialize values
clear;clc;
TR = 50;
T1 = 1000;
T2_dephasing = 50;
n=6;
Alpha =
linspace(pi/12,pi/2,n);
```

```
% Iteration: N times of RF
N = 10;
% Mz
M_decade = ones(N,n);
M_recovery=[];
Mz=ones(1,n);
% Mxy
Mxy_decade = [];
Mxy_recovery=zeros(N,n);
Mxy=zeros(1,n);
```

Method

Iteration

```
for i=1:N
    % Excitation
    if i>1
        M_decade(i,:) =
            Mz(end,:).*cos(Alpha);
        Mxy_recovery(i,:) =
            Mz(end,:).*sin(Alpha);
    else
        M_decade(i,:) =
            ones(1,n).*cos(Alpha);
        Mxy_recovery(i,:) =
            ones(1,n).*sin(Alpha);
    end
    Mz = [Mz; M_decade(i,:)];
    Mxy = [Mxy; Mxy_recovery(i,:)];
end
```

```
% Relaxation
for j=2:TR
    M_recovery =
        [M_recovery; 1+(M_decade(i,:)-
            1).*exp(-(j)/T1)];
    Mz = [Mz; M_recovery(end,:)];
    Mxy_decade = [Mxy_decade;
        Mxy_recovery(i,:).*exp(-
            (j)/T2_dephasing)];
    Mxy = [Mxy; Mxy_decade(end,:)];
end
end
```

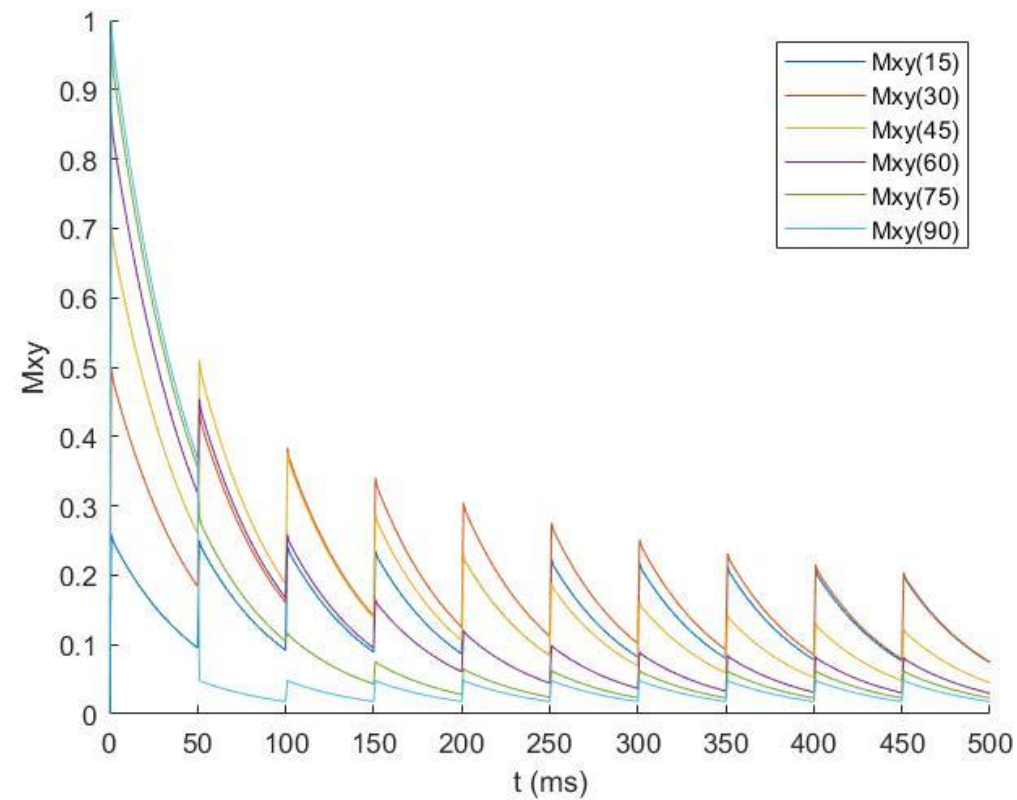
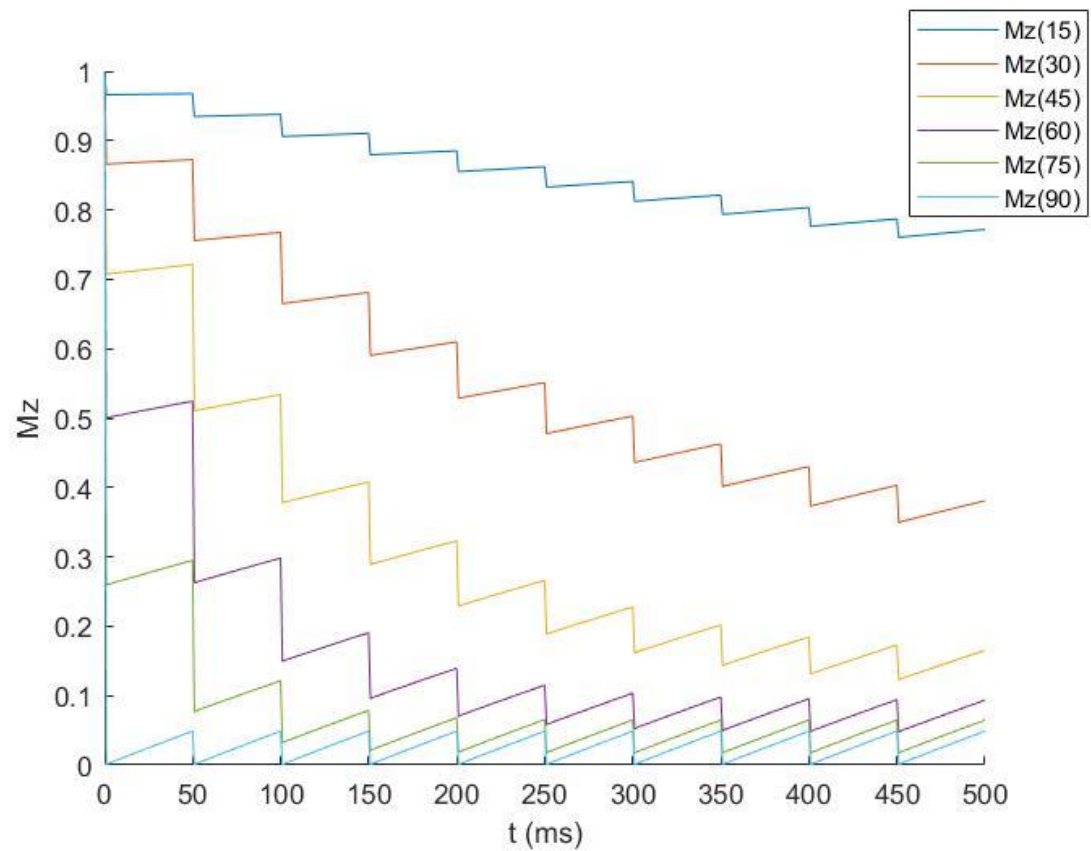
Method

Plot

```
% Plot Mz
T = 0:TR*N;
figure();
hold on;
for i=1:n
    plot(T,Mz(:,i));
end
LineName =
    ['Mz(15)'; 'Mz(30)'; 'Mz(45)';
    'Mz(60)'; 'Mz(75)'; 'Mz(90)'] ;
legend(LineName);
axis([min(T(:)) max(T(:)) 0 1]);
ylabel('Mz'); xlabel('t (ms)')
```

```
% Plot Mxy
T = 0:TR*N;
figure();
hold on;
for i=1:n
    plot(T,Mxy(:,i));
end
LineName =
    ['Mxy(15)'; 'Mxy(30)'; 'Mxy(45)';
    'Mxy(60)'; 'Mxy(75)'; 'Mxy(90)'] ;
legend(LineName);
axis([min(T(:)) max(T(:)) 0 1]);
ylabel('Mxy'); xlabel('t (ms)')
```


Experimental Results



OUTLINE

- EX2.1 Simulate The Timing of M_{xy} and M_z
 - Introduction
 - Method
 - Experimental Results
- EX2.2 The Maximum MRI Image Brightness
 - Introduction
 - Method
 - Experimental Results
- EX2.3 Subcortical Structures
 - Introduction
 - Method
 - Experimental Results

事實上信號強度的公式是

- 信號正比於

$$\frac{(1 - e^{-TR/T1}) \sin \alpha}{1 - e^{-TR/T1} \cos \alpha} e^{-TE/T2^*}$$

- α : 偏折角

Initialize & Calculate

```
% Initialize values
```

```
clear;clc;
```

```
TR = 100;
```

```
T2_dephasing = 50;
```

```
n=901;
```

```
Alpha = linspace(0,pi/2,n);
```

```
% Calculate Mz
```

```
T1 = [1000 1100]
```

```
% for T1_1
```

```
Mz(1,:) = ((1-exp(-TR/T1(1))).*sin(Alpha))./(1-exp(-  
TR/T1(1)).*cos(Alpha));
```

```
% for T1_2
```

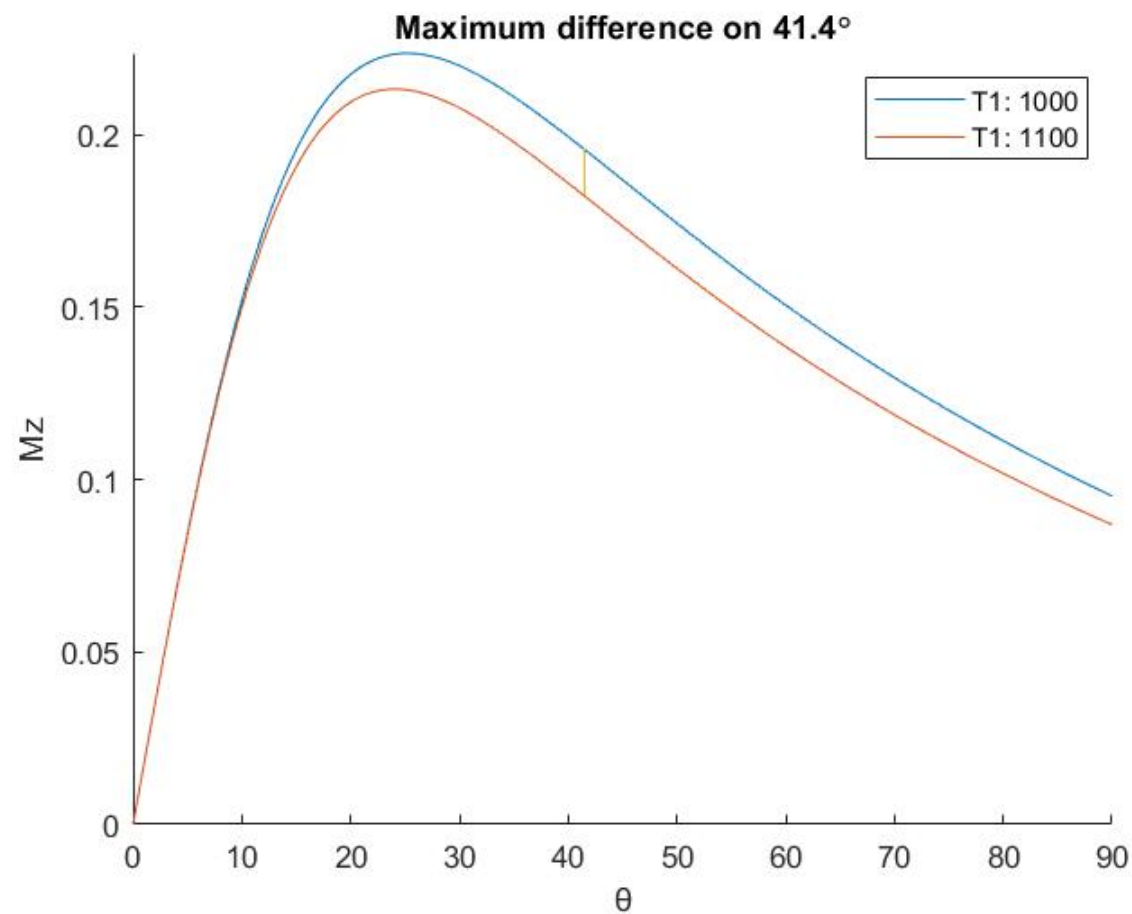
```
Mz(2,:) = ((1-exp(-TR/T1(2))).*sin(Alpha))./(1-exp(-  
TR/T1(2)).*cos(Alpha));
```

Find the Different between two number

```
% Find maximum MRI image brightness
difference between the two tissues(T1s)
different = abs(Mz(2,:)-Mz(1,:));
index = find(different ==
max(different));
maxAngle = Alpha(index)*180/pi
```

```
% Show the result
figure();
hold on
plot(Alpha*180/pi,Mz(1,:))
plot(Alpha*180/pi,Mz(2,:))
plot([index/10 index/10],Mz(:,index))
txt = ['Maximum difference on ',
       num2str(maxAngle), '\circ'];
xlabel("\theta")
ylabel("Mz")
title(txt)
legend([strcat("T1: ",num2str(T1(1))),
       strcat("T1: ",num2str(T1(2)))])
axis([0 90 min(Mz(:)) max(Mz(:))])
```

Experimental Results



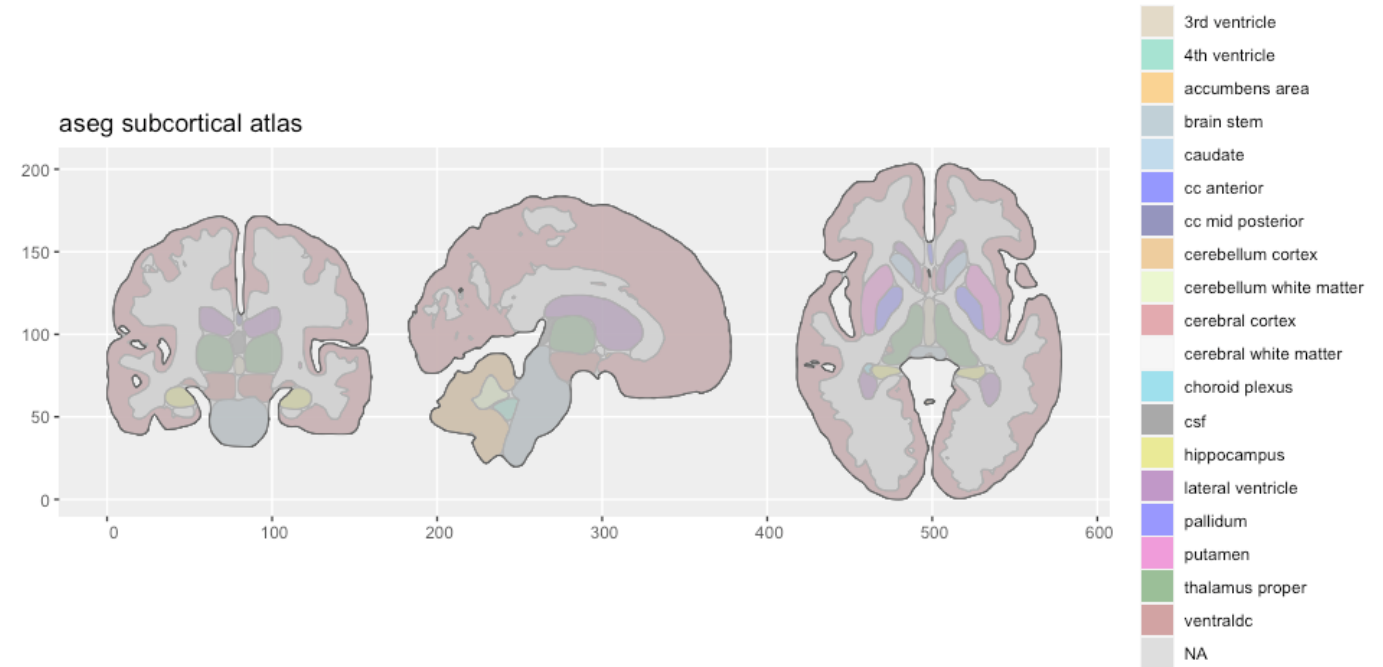
OUTLINE

- EX2.1 Simulate The Timing of M_{xy} and M_z
 - Introduction
 - Method
 - Experimental Results
- EX2.2 The Maximum MRI Image Brightness
 - Introduction
 - Method
 - Experimental Results
- EX2.3 Subcortical Structures
 - Introduction
 - Method
 - Experimental Results

Introduction

The subcortical

1. Read the `nil.gz` of MRI files.
2. Calculate volumes of each subcortical area.
3. Read the `IXI.xls` by pandas dataform.
4. Correlation analysis between volume and personal informations.



Method

Import

```
import nibabel as nib
import pandas as pd
import numpy as np
import os
import glob
import seaborn as sns
import matplotlib.pyplot as plt
```

- nibabel: to read nii file
- pandas: to data analysis
- numpy: for calculate
- os: write and read the file
- glob: read the path of file
- matplotlib and seaborn: plot the result

Method

Initialize

```
# setting the path of files
files = glob.glob('IXI_aseg/*.nii.gz')

# create list to save labels of each area
labels=[4,5,7,8,10,11,12,13,14,15,16,17,18,
        24,26,28,30,31,43,44,46,47,49,50,51,
        52,53,54,58,60,62,63,72,77,78,79,80,
        81,82,85,251,252,253,254,255]
```

Label No.	Structure Name	Label No.	Structure Name
4	Left-Lateral-Ventricle	50	Right-Caudate
5	Left-Inf-Lat-Vent	51	Right-Putamen
7	Left-Cerebellum-White-Matter	52	Right-Pallidum
8	Left-Cerebellum-Cortex	53	Right-Hippocampus
10	Left-Thalamus-Proper	54	Right-Amygdala
11	Left-Caudate	58	Right-Accumbens-area
12	Left-Putamen	60	Right-VentralDC
13	Left-Pallidum	62	Right-vessel
14	3rd-Ventricle	63	Right-choroid-plexus
15	4th-Ventricle	72	5th-Ventricle
16	Brain-Stem	77	WM-hypointensities
17	Left-Hippocampus	78	Left-WM-hypointensities
18	Left-Amygdala	79	Right-WM-hypointensities
24	CSF	80	non-WM-hypointensities
26	Left-Accumbens-area	81	Left-non-WM-hypointensities
28	Left-VentralDC	82	Right-non-WM-hypointensities
30	Left-vessel	85	Optic-Chiasm
31	Left-choroid-plexus	251	CC_Posterior
43	Right-Lateral-Ventricle	252	CC_Mid_Posterior
44	Right-Inf-Lat-Vent	253	CC_Central
46	Right-Cerebellum-White-Matter	254	CC_Mid_Anterior
47	Right-Cerebellum-Cortex	255	CC_Anterior
49	Right-Thalamus-Proper		

Method

Calculate

```
# calculate volumes of each area
if not os.path.exists('vol_all.npy'):
    vol_all = []
    for file in files:
        vol = []
        img = nib.load(file)
        img_array = img.get_fdata()
        factor = np.prod(img.header.get_zooms())
        for label in labels:
            vol.append(np.sum(img_array==label)*factor)
        vol_all.append(vol)
    # save result to `*.npy`
    np.save('vol_all', vol_all)
# if done, load the `*.npy`
else:
    vol_all = np.load('vol_all.npy')
```

Method

Dataform

```
# create list to save the ID of each nii file.
id = [int(os.path.basename(file).split('-')[0][3:6]) for file in files]
# read xls file by pandas
df = pd.read_excel('IXI.xls')
# change index
df.index = df['IXI_ID']

# match ID between nii file and xls
id_new = [i for i in id if i in df['IXI_ID']]
# create new df
df_new = df.loc[id_new]
df_new.drop_duplicates(subset='IXI_ID', keep='first', inplace=True)
# save new df
df_new.to_excel('df_new.xlsx')
```

Method

Dataform

```
# create df of volumes
vol_all = np.array(vol_all)
df_vol = pd.DataFrame(vol_all, columns = labels)
df_vol.insert(loc=0, column="IXI_ID", value=id)
df_vol.index = df_vol['IXI_ID']

# match ID between nii file and xls
df_vol_new = df_vol.loc[id_new]
df_vol_new.drop_duplicates(subset='IXI_ID', keep='first', inplace=True)
df_vol_new.to_excel('df_vol_new.xlsx')
```

Method

Dataform

```
# concat df and df_vol
df_concat = pd.concat([df_new, df_vol_new.drop(['IXI_ID'], axis=1)], axis=1)
df_concat.to_excel('df_concat.xlsx')

# drop some information we don't use: 'IXI_ID', 'DOB', 'DATE_AVAILABLE', 'STUDY_DATE'
df_concat = df_concat.drop(['IXI_ID', 'DOB', 'DATE_AVAILABLE', 'STUDY_DATE'], axis=1)
# Fill 0 to Nan value
df_concat.fillna(0)
# change 0 to mean value
df_concat['HEIGHT'] = df_concat['HEIGHT'].replace(0, df_concat['HEIGHT'].mean())
df_concat['WEIGHT'] = df_concat['WEIGHT'].replace(0, df_concat['WEIGHT'].mean())
df_concat['AGE'] = df_concat['AGE'].replace(0, df_concat['AGE'].mean())
# drop some missing data
df_concat = df_concat[df_concat['ETHNIC_ID'] != 0]
df_concat = df_concat[df_concat['MARITAL_ID'] != 0]
df_concat = df_concat[df_concat['OCCUPATION_ID'] != 0]
df_concat = df_concat[df_concat['QUALIFICATION_ID'] != 0]
```

Method

Dataform

	SEX_ID (1=m, 2=f)	HEIGHT	WEIGHT	ETHNIC_ID	MARITAL_ID	OCCUPATION_ID	QUALIFICATION_ID	AGE	4	5	...	79	80	81	82	85	251	252
IXI_ID																		
2	2	164.0	58.0	1	4	1	5	35.800137	4520.379385	167.694896	...	0.0	0.0	0.0	0.0	67.499832	1066.286411	762.537166
12	1	175.0	70.0	1	2	1	5	38.781656	9688.387846	129.726944	...	0.0	0.0	0.0	0.0	205.664667	933.401181	591.681426
13	1	182.0	70.0	1	2	1	5	46.710472	7583.197982	91.757750	...	0.0	0.0	0.0	0.0	82.265569	1008.280566	681.327663
14	2	163.0	65.0	1	4	1	5	34.236824	3846.445747	84.375010	...	0.0	0.0	0.0	0.0	13.710939	1021.992303	651.796949
15	1	181.0	90.0	2	1	6	5	24.284736	5560.327583	59.062660	...	0.0	0.0	0.0	0.0	82.265848	1485.004028	861.682025
...
648	1	193.0	120.0	1	1	6	4	47.723477	12749.104289	257.344594	...	0.0	0.0	0.0	0.0	202.500664	1152.777216	781.525999
651	1	175.0	61.0	3	2	8	2	50.395619	2762.226250	132.890610	...	0.0	0.0	0.0	0.0	143.437484	975.585827	553.710875
652	1	163.0	80.0	1	1	1	5	42.989733	7538.916475	111.797027	...	0.0	0.0	0.0	0.0	121.289227	1352.111209	786.797942
653	1	172.0	100.0	1	3	1	5	46.220397	5079.384760	258.398934	...	0.0	0.0	0.0	0.0	156.094050	1219.221093	634.923095
662	1	182.0	98.0	1	4	1	3	41.741273	8095.806871	473.556186	...	0.0	0.0	0.0	0.0	209.883477	993.518769	561.095526
471 rows × 53 columns																		

Correlation Analysis

```
# correlation analysis
for lebel in labels:
    df_corr = df_concat.iloc[:,0:8]
    corr_label = lebel
    df_corr[lebel]=df_concat[lebel]

# compute pairwise correlation of columns by df.corr
corr = df_corr.corr()
corr_result = corr[lebel]
```

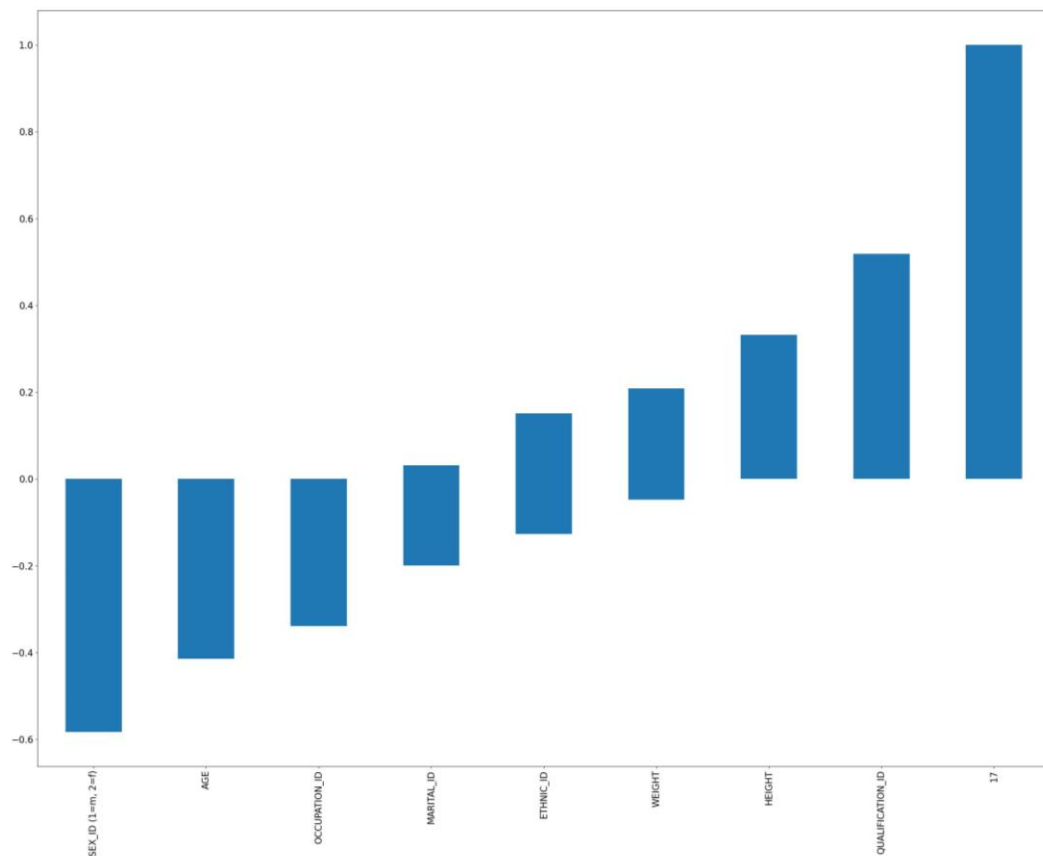

Method

Plot

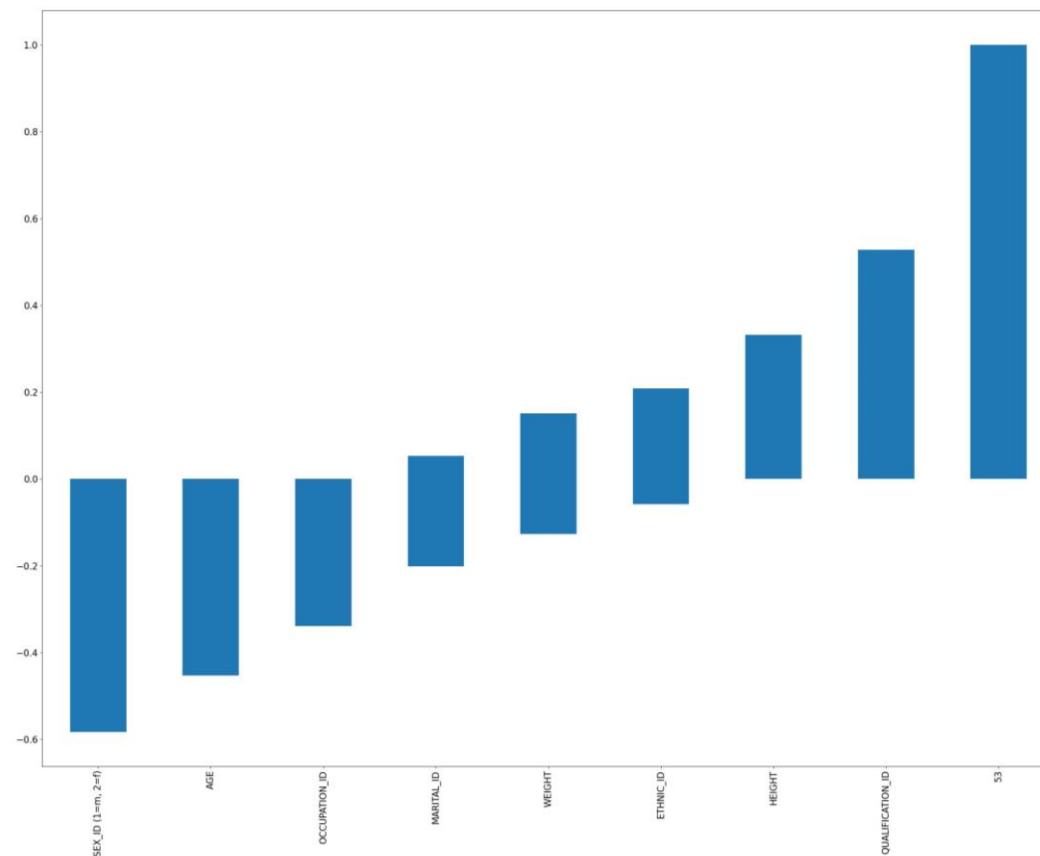
```
# plot the result bar
fig = plt.figure(figsize=(6,4))
corr_result.sort_values().plot(kind='bar',figsize=(32, 24), fontsize=24).get_figure()
# save the result of bar
if not os.path.exists('images/result/'):
    os.mkdir('images/result')
fig.savefig('images/result/IXI_Volume_'+str(lebel)+'.jpg')
# clear fig
plt.cla()
plt.clf()
# plot the result by heatmap
sns.heatmap(corr, vmin=-1, vmax=1, annot=True, linecolor="white")
# save the result of heatmap
fig.savefig('images/result/IXI_Volume_heatmap_'+str(lebel)+'.jpg')
# close fig
plt.close()
```

Experimental Results

Left-Hippocampus

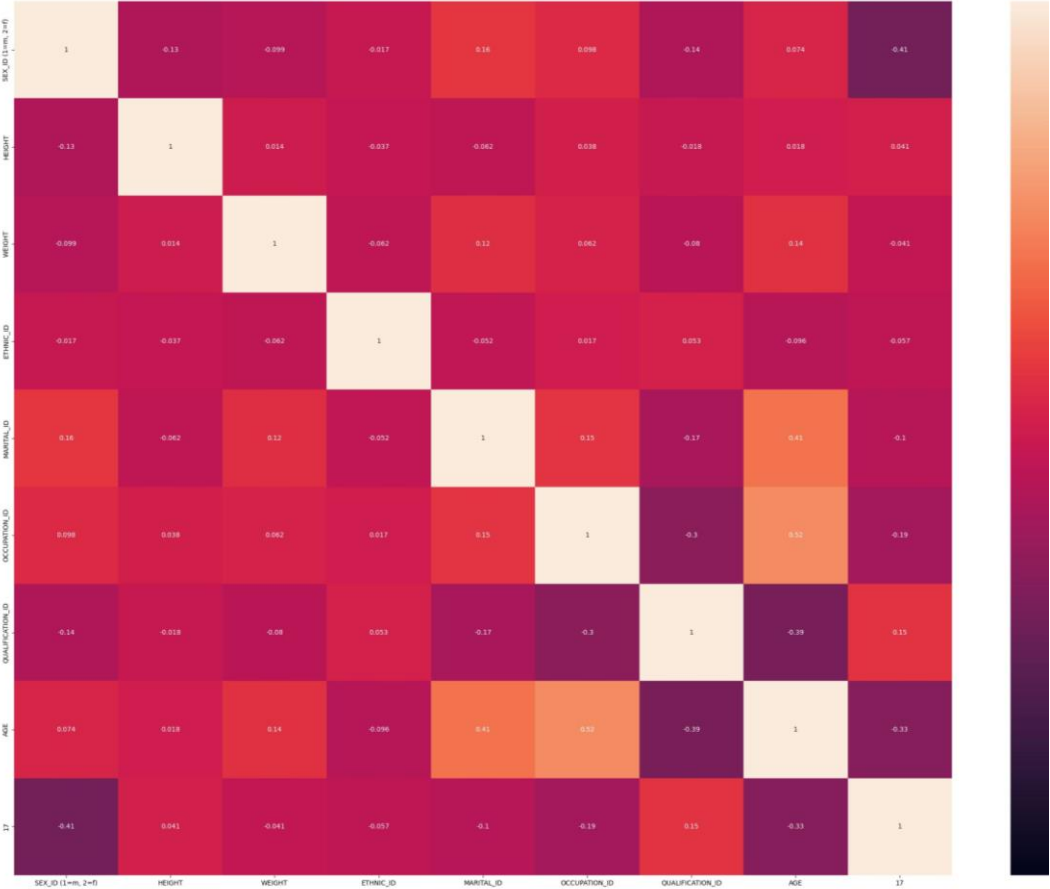


Right-Hippocampus



Experimental Results

Left-Hippocampus



Right-Hippocampus

