**MATLAB Basics for Psych 4450 - Part 2**

Prepared by Aahana Bajracharya ([aahana@wustl.edu](mailto:aahana@wustl.edu))

If you have any questions about MATLAB or the class, feel free to email me, post to the #matlab channel on Slack (everyone can benefit!), direct message me on Slack, or set up a time here <https://calendly.com/aahanabajra> to chat via Zoom.

If you run into issues or have specific questions about this document, you can also make a comment and I will try to update it to address your question.

In this part of the MATLAB Basics guide, we will focus on using MATLAB to navigate neuroimaging data and using some built-in SPM functions. Refer to **part 1** for an introduction.

## 1.0 Reading and Plotting NIfTI files

NIfTI stands for Neuroimaging Informatics Technology Initiative and is the name given to the most common type of neuroimaging data file.

The code for the material below is available in Canvas as **read\_plot\_nifti.m.**

The path below is pointing to the location of the folder containing the subject of interest. Change it to the path in YOUR computer. If you have a mac, it'll probably look something like this: /Users/aahana/Documents/MATLAB/Dataset

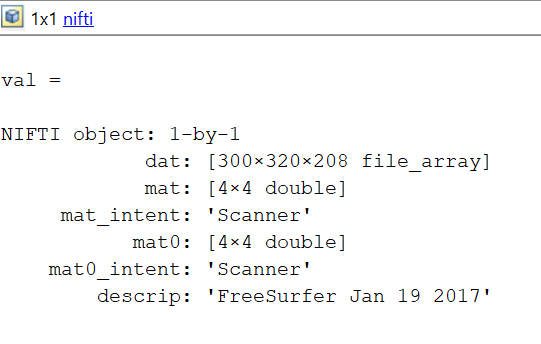
yourPath = 'C:\Users\aahan\Desktop\Psych4450\ds002382\_fMRI\_data\_partial\young\';

We will be using the nifti command to load the image file into MATLAB. nifti.m is built into SPM, so you need to have SPM in your path. Refer to Psych 4450 MATLAB basics guide part 1 for info on setting paths.

sub01\_T1 = nifti([yourPath 'sub-01\anat\sub-01\_T1w.nii']);

% instead of creating the variable yourPath, you can also directly enter the full path to the nifti

The information from our file is now in the nifti object sub01\_T1. Click on **1x1 nifti** on the Workspace to see what's inside it. You will see the following.



The NIFTI object consists of various subcategories of information. For our purpose, we want to extract the information contained in the **dat** variable to a matrix. We do this by typing **sub01\_T1.dat.** Let’s say if you were interested to obtain information from **mat0** variable in the above figure, you would type **sub01\_T1.mat0**. Think of the (.) dot as a way of accessing the subcategories of information in the NIFTI object.

sub01\_data = sub01\_T1.dat(:,:,:);

% the colons inside the parenthesis represents the 3 dimensions of the data

% Let's check what's in voxel (20,20,1)

sub01\_data(20,20,1) % the output is the voxel intensity, ans = 16.1765

**imagesc** is used to display images with scaled colors. We have to use the **squeeze ()** function to get **imagesc ()** to work properly. **squeeze ()** function changes the size of the multidimensional array to the one you specify. Refer to the end of this guide for more info on **squeeze** function.

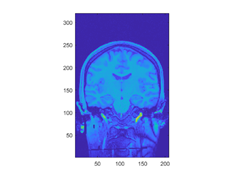
% Let's check what's in plane x =150 (aka a brain slice with x fixed at 150)

figure; % Coronal slice

imagesc(squeeze(sub01\_data(150,:,:)))

axis image; % this makes sure MATLAB doesn't stretch the figure

set(gca,'YDir','normal');% this sets the coordinate plane to start at bottomleft

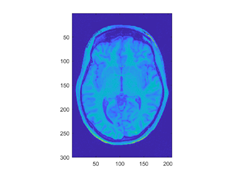


% Let's check what's in plane y =150 (aka a brain slice with y fixed at 150)

figure; % Axial/Horizontal/Transverse slice

imagesc(squeeze(sub01\_data(:,150,:)))

axis image;



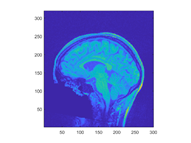
% Let's check what's in plane z =98 (aka a brain slice with z fixed at 98)

figure; % Sagittal slice

imagesc(squeeze(sub01\_data(:,:,98))') % notice the transpose sign (‘). This was done to match the SPM output.

axis image;

set(gca,'YDir','normal');



We can also plot all these in the same figure using **subplot (m,n,p)**. A single figure is divided into m rows and n columns to accommodate for subplots. p denotes the position of the plot.

figure;

% Coronal slice

subplot(1,3,1) % Figure is divided into 1 row and 3 columns and the image will be placed in the top position

imagesc(squeeze(sub01\_data(150,:,:)))

axis image;

set(gca,'YDir','normal');

% Axial/Horizontal/Transverse slice

subplot(1,3,2)

imagesc(squeeze(sub01\_data(:,150,:)))

axis image;

set(gca, 'XAxisLocation', 'top');

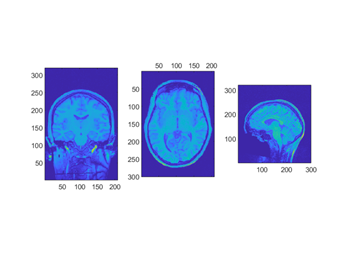
% Sagittal slice

subplot(1,3,3)

imagesc(squeeze(sub01\_data(:,:,98))')

axis image;

set(gca,'YDir','normal');



Let's plot a montage of slices using a loop. The subplots first fill the rows and then the columns.

In the code chunk below, **num\_slices** variable is a vector with the slice numbers that you want to plot. Instead of using the colon (:) to create the vector, you can also write out the list of slices yourself. For example: **num\_slices = [2,3,5,89,95]**. This way you can choose slices at irregular intervals. Doing this **num\_slices = [50,70,90,110,130,150,170,190,210,230,250]** is the same as the way we have generated the vector below.

**numel(num\_slices) = 11**, which means that the vector num\_slices has 11 elements in it. In the for loop, we will be iterating over these 11 elements.

The **figure** command creates an empty figure as a placeholder. The subplot command then divides this empty placeholder figure into the specified number of subfigures which can be populated individually. In our case, **subplot(4,3, i)** creates divided the figure into 4 rows and 3 columns (Example: the grid below). I set it to this because we have 11 brain slices to plot. Here, **i** would represent the position of the figures to be populated based on the iteration number. This is the order in which the slices will be plotted (along the row first). In the first iteration of the loop, **i = 1**. So, **subplot(4,3,1)** would fill the 1X1 position of the placeholder figure.

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |
| 10 | 11 |  |

figure; % creates an empty figure as a placeholder

num\_slices= 50:20:250; % Choosing slices that start at 50 and end at 250 with intervals of 20

for i= 1:numel(num\_slices) % numel counts the number of elements in num\_slices

subplot(4,3,i) % starts filling the empty figure

imagesc(squeeze(sub01\_data(num\_slices(i),:,:)));

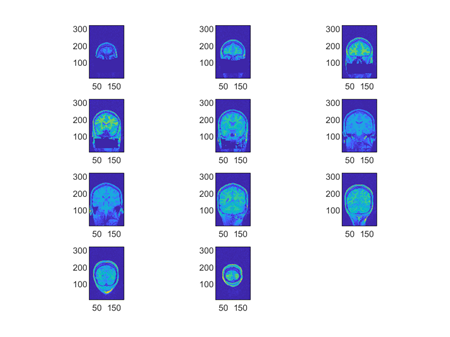
% by doing num\_slices(i) , we are indexing over each of the elements in num\_slices

% Try changing the position of num\_slices(i) to y or z position

axis image; % making sure MATLAB doesn’t stretch the image

set(gca,'YDir','normal'); % flipping the image to match SPM output

end

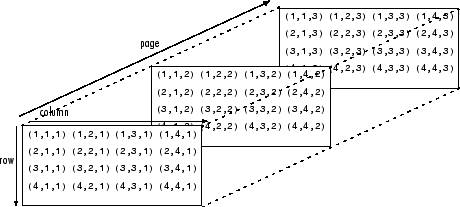


**Helpful resources:**

<https://www.fmrwhy.com/2018/06/28/spm12-matlab-scripting-tutorial-1/>

**Squeeze** function

Our 3D fMRI data looks like the following. But when plotting slices, we are essentially choosing only one of these “pages” from the stack of 2D matrices below. So, the **squeeze** function is helping us to fixate on a specific “page” or slice in our case.



Here’s a simpler example:

A(:,:,1) =

1

2

A(:,:,2) =

3

4

B = squeeze(A)

B = 2×2

1 3

2 4

Here, A is a 2X1X2 array and we are removing the dimension of length 1, which results in a 2x2 matrix.