Functions

PNI Summer Internship 2020

Mai Nguyen

Outline

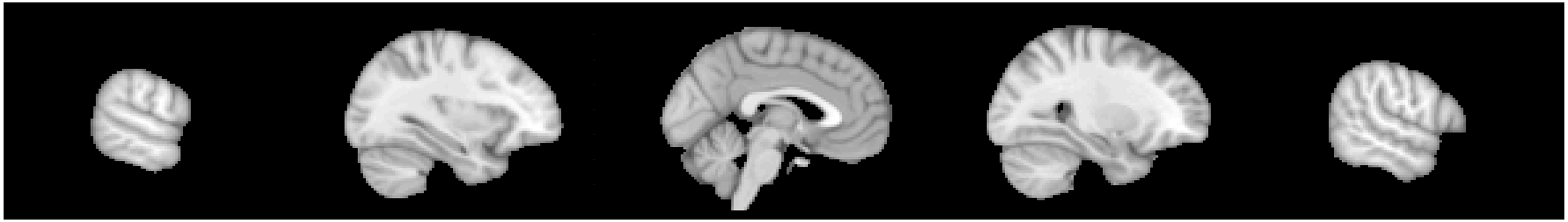
- Review HW 1
- What are functions?
- Write your first function
- Syntax & best practices
- Outputs and scope
- Why use functions?
- Quickstart debugging

HW 1

- Solutions posted after class
- Common errors + suggestions:
    - Selecting slice of 3D data but getting 3D matrix instead of 2D -> use squeeze( )
    - Giving a variable the same name as a built-in function
    - Practice white space and documentation
    - Specifying the dimension for an operation (e.g. mean)

What are functions?

- In HW 1, you visualized slices of the brain. You can change slice by changing the index:



index = 15 **index = 30** **index = 45** **index = 60** **index = 75**

- Annoying: have to copy/paste lines of code over and over to do the same thing over and over with slightly different numbers

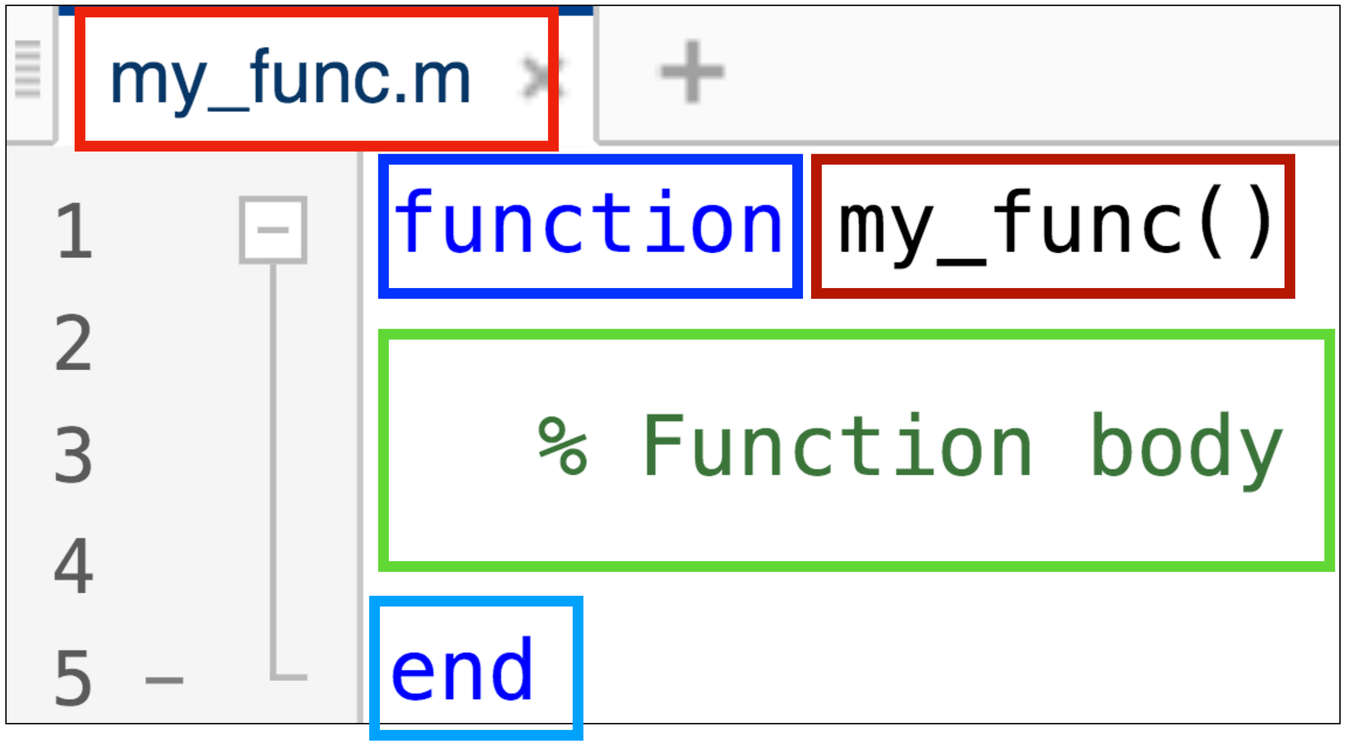
What are functions?

- Block of re-usable code that does performs same task on different inputs (shortcut)
- Many built-in functions:
    - rand(m,n), randi(a, m, n)
    - size(matrix), length(matrix)
    - imagesc(matrix)
    - disp(val)
    - cat(dim, mat1, mat2), horcat(mat1, mat2), vertcat(mat1, mat2)

Write your first function

- hello(name)

Writing your own functions: syntax



- Start with keyword **function**
- **Function name** followed by parenthesis. Follows same rules as variable naming
- **File name** is same as function name
- Function definition closes with keyword **end**
- **Function body** in the middle

Writing your own functions: syntax

my_func.m

```
1  function my_func(arg1)
2
3      % Function body
4
5  end
```

- Add **arguments** (inputs) in parentheses
- Arguments make functions flexible: do the same operations on different values

Writing your own functions: syntax

my_func.m

```
1  function my_func(arg1, arg2, arg3)
2
3      % Function body
4
5  end
```

- Add **arguments** (inputs) in parentheses
- Arguments make functions flexible: do the same operations on different values
- Separate multiple arguments with commas

Write your second function

- hello(name)
- calc_area(width, length)

Write your own functions: practice!

- Go back to your script from HW 1. We'll modify the script for visualizing brain slices into a function
- Write a function called plot_sagittal_slice( )
    - One argument: slice index
    - For that index, get the sagittal brain slice and store to a variable
    - Plot sagittal brain slice
    - Test your function by calling:
        - plot_sagittal_slice(25)
        - plot_sagittal_slice(50)
        - plot_sagittal_slice(60)

Write your own functions: outputs and scope

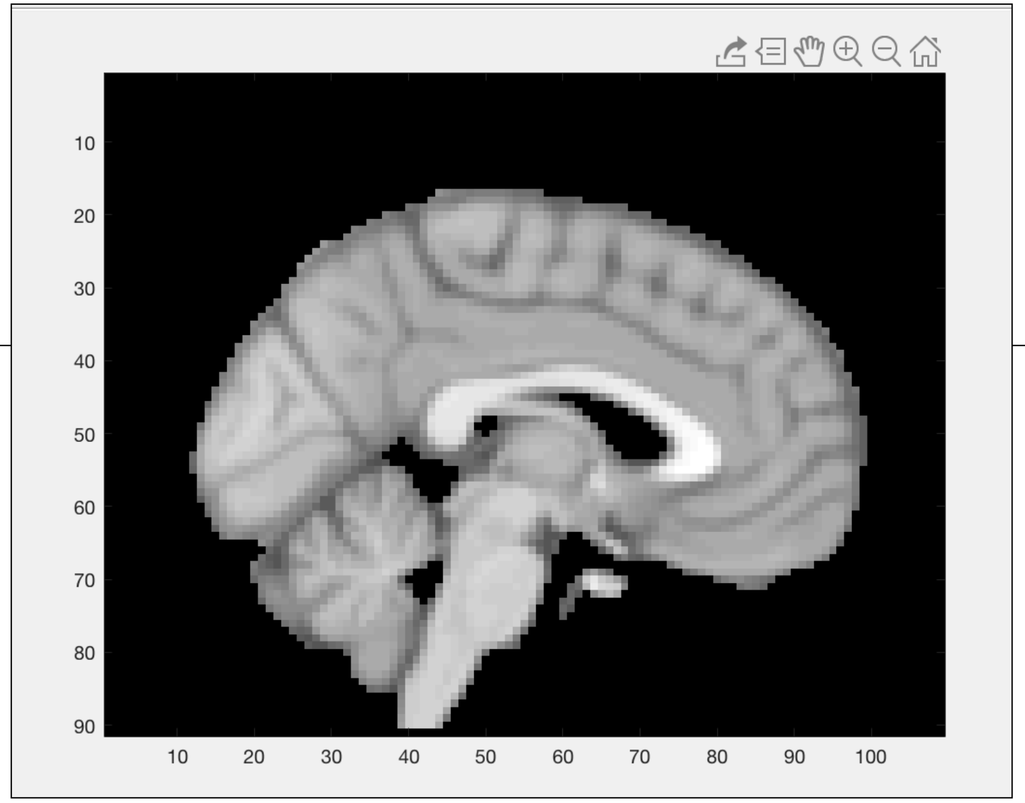
Editor - /Users/Mai/Projects/teaching/matlab/week2/...

week2_lec3_script.m | plot_saggital_slice.m

```
1   function plot_saggital_slice(ind)
2
3   % load data
4   load('MNI152_T1_2mm_brain.mat');
5
6   % get sagittal slice
7   sag_slice = squeeze(nifti_img(ind,:,:));
8
9   % plot
10  figure; colormap gray;
11  imagesc(flipud(sag_slice'));
```

Command Window

```
>> plot_saggital_slice(45)
```



Write your own functions: outputs and scope

Editor - /Users/Mai/Projects/teaching/matlab/week2/...

week2_lec3_script.m | plot_saggital_slice.m

```
1   function plot_saggital_slice(ind)
2
3   % load data
4   load('MNI152_T1_2mm_brain.mat');
5
6   % get sagittal slice
7   sag_slice = squeeze(nifti_img(ind,:,:));
8
9   % plot
10  figure; colormap gray;
11  imagesc(flipud(sag_slice'));
```

Command Window

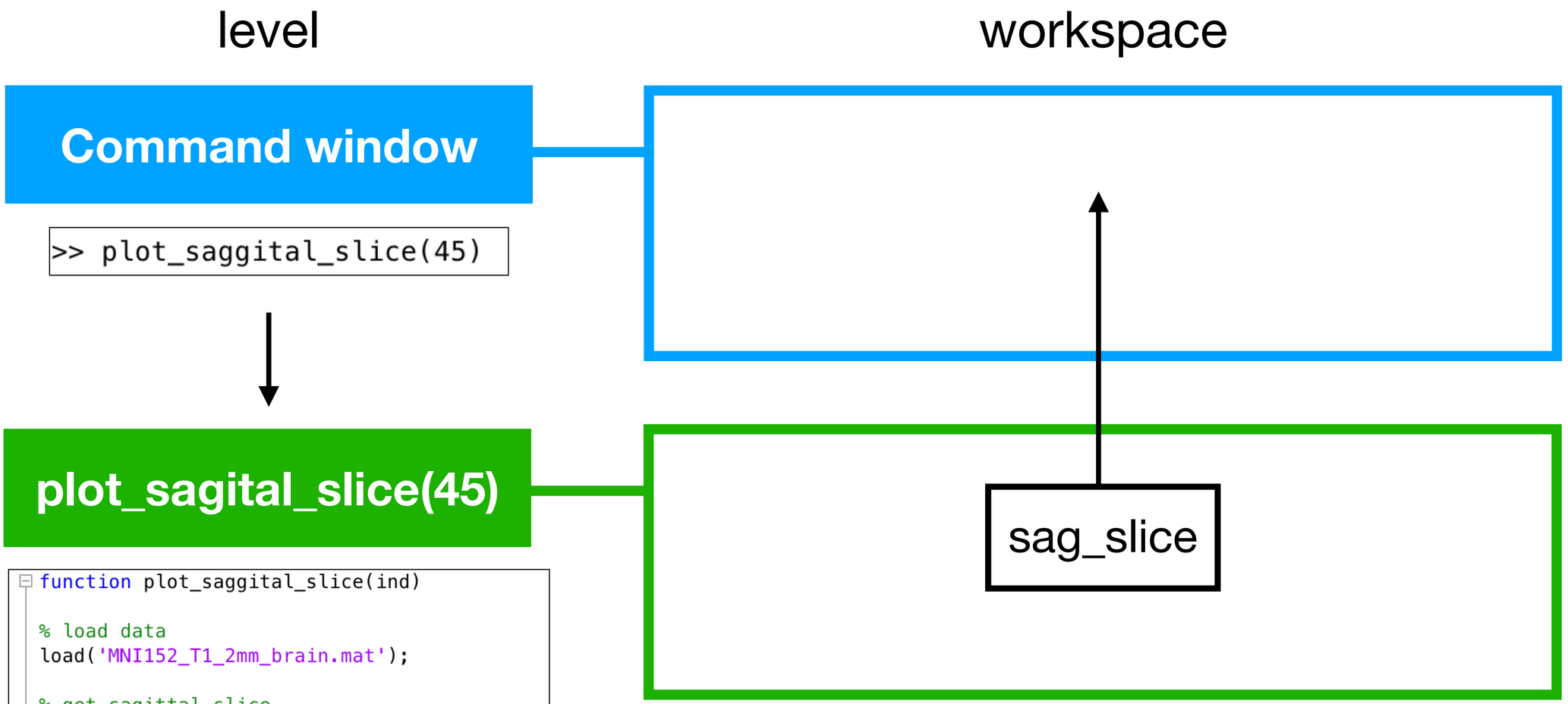
```
>> plot_saggital_slice(45)
>> sag_slice
Undefined function or variable 'sag_slice'.

fx >>
```

Functions have local scope

- “sag_slice” is a local variable: only defined within the function
- command window doesn’t know what happens inside plot_sagittal_slice

Write your own functions: outputs and scope




```
function plot_saggital_slice(ind)

% load data
load('MNI152_T1_2mm_brain.mat');

% get sagittal slice
sag_slice = squeeze(nifti_img(ind,:,:));

% plot
figure; colormap gray;
imagesc(flipud(sag_slice'));
```

Write your own functions: outputs and scope

week2_lec3_script.m | plot_saggital_slice.m

```
1   function sag_slice = plot_saggital_slice(ind)
2
3   % load data
4   load('MNI152_T1_2mm_brain.mat');
5
6   % get sagittal slice
7   sag_slice = squeeze(nifti_img(ind,:,:));
8
9   % plot
10  figure; colormap gray;
11  imagesc(flipud(sag_slice'));
```

Command Window

```
>> my_slice = plot_saggital_slice(45);
>> size(my_slice)

ans =

   109    91
```

- Add **outputs** before function name
- Outputs lets you take values from inside the function to outside

Writing your own functions: syntax

my_func.m

```
1   function [out1, out2] = my_func(arg1, arg2, arg3)
2
3       % Function body
4
5   end
```

- Add **outputs** before function name
- Outputs lets you take values from inside the function to outside
- Can have multiple outputs: separate by commas and surround [ ]

Write your own functions: practice 2!

- Write a function to calculate the circumference and area of a circle from a radius r
    - Takes one input, radius
    - Outputs two values, area and circumference
- Test your function:
    - calc_circle(1)
    - calc_circle(.5)
    - calc_circle(66)

Why use functions?

- **Generalizability**: call a function multiple times with different inputs
- **Reusability:** can use same function in multiple contexts
- **Prevents errors:** not typing/editing in line prevents error
- **Easier to change**: only need to change in one place
- **Breaks problems into smaller chunks**

Why use functions

```
% Demo analysis script

%% Load subject data:



%% Threshold



%% Normalize



%% Run stats
```

- Analysis script break down analysis problem into smaller chunks

Why use functions

```
1
2   % Demo analysis script
3
4   %% Load subject data:
5   % -function that gets data for all subjects from .csv files and
6   % organizes in a 3D matrix (nSubs x nVoxels x nTRs
7
8   %% Threshold
9   % function that sets min threshold and removes subject data below threshold
10
11
12
13
14
15  %% Normalize
16  % function to normalize data
17
18
19
20
21  %% Run stats
22  % function to do statistics
23
24
25
26
```

- Analysis script break down analysis problem into smaller chunks
- Functions for each chunk

Why use functions

```
% Demo analysis script

%% Load subject data:
% -function that gets data for all subjects from .csv files and
% organizes in a 3D matrix (nSubs x nVoxels x nTRs

%% Threshold
% function that sets min threshold and removes subject data below threshold
% Sub functions:
%   -function to determine threshold
%   -function to apply threshold
%   -function to save data

%% Normalize
% function to normalize data
% Sub functions:
%   -function to apply normalization
%   -function to save data

%% Run stats
% function to do statistics
% Sub functions:
%   -function to run stat test
%   -function for multiple comparisons corrections
%   -function for calculating p-values
```

- Analysis script break down analysis problem into smaller chunks
- Functions for each chunk
- Functions may even have sub-functions that further break down the analysis
- Code modularity: each function only does 1 thing

Write your own functions: best practices

- Write a function if you find yourself running the same lines of code multiple times
- Simple, does one thing (ish)
- Descriptive function names, with descriptive arguments and outputs
- DOCUMENTATION! After the function header:
    - Copy function header in a comment
    - Describe what the function does
    - Describe arguments and inputs, including datatype and dimensions if applicable
    - Extra: Examples of how to use

Example good documentation

week2_lec3_script.m | plot_saggital_slice.m

```
1   function sag_slice = plot_saggital_slice(ind)
2   % function sag_slice = plot_saggital_slice(ind)
3   %
4   % Plots the sagittal slice specified by ind from the MNI152 2mm brain. The
5   % brain imagae must be in the current directory.
6   %
7   % ARGUMENTS:
8   % -ind: integer, specifies slice to be plotted. Range between 1-91
9   %
10  % OUTPUTS:
11  % -sag_slice: 109x91 matrix containing MNI data for the specified slice
12  %
13  % EXAMPLE USAGE:
14  % -my_slice = plot_saggital_slice(45);
15
16
17  % load data
18  load('MNI152_T1_2mm_brain.mat');                          %#ok<*LOAD>
19
20  % get sagittal slice
21  sag_slice = squeeze(nifti_img(ind,:,:));                  %#ok<*NODEF>
22
23  % plot
24  figure; colormap gray;
25  imagesc(flipud(sag_slice'));
```

Command Window

```
>> help plot_saggital_slice
  function sag_slice = plot_saggital_slice(ind)

  Plots the sagittal slice specified by ind from the MNI152 2mm brain. The
  brain imagae must be in the current directory.

  ARGUMENTS:
  -ind: integer, specifies slice to be plotted. Range between 1-91

  OUTPUTS:
  -sag_slice: 109x91 matrix containing MNI data for the specified slice

  EXAMPLE USAGE:
  -my_slice = plot_saggital_slice(45);
```

Quickstart debugging

Grace Hopper, one of the earliest and most influential modern computer programmer

Use error messages

Editor – /Users/Mai/Projects/teaching/matlab/week2/plot_sa...

week2_lec3_script.m | plot_saggital_slice.m

```
1   function sag_slice = plot_saggital_slice(ind)
2
3   % load data
4   load('MNI152_T1_2mm_brain.mat!@!@');
5
6   % get sagittal slice
7   sag_slice = squeeze(nifti_img(ind,:,:));
8
9   % plot
10  figure; colormap gray;
11  imagesc(flipud(sag_slice'));
```

Command Window

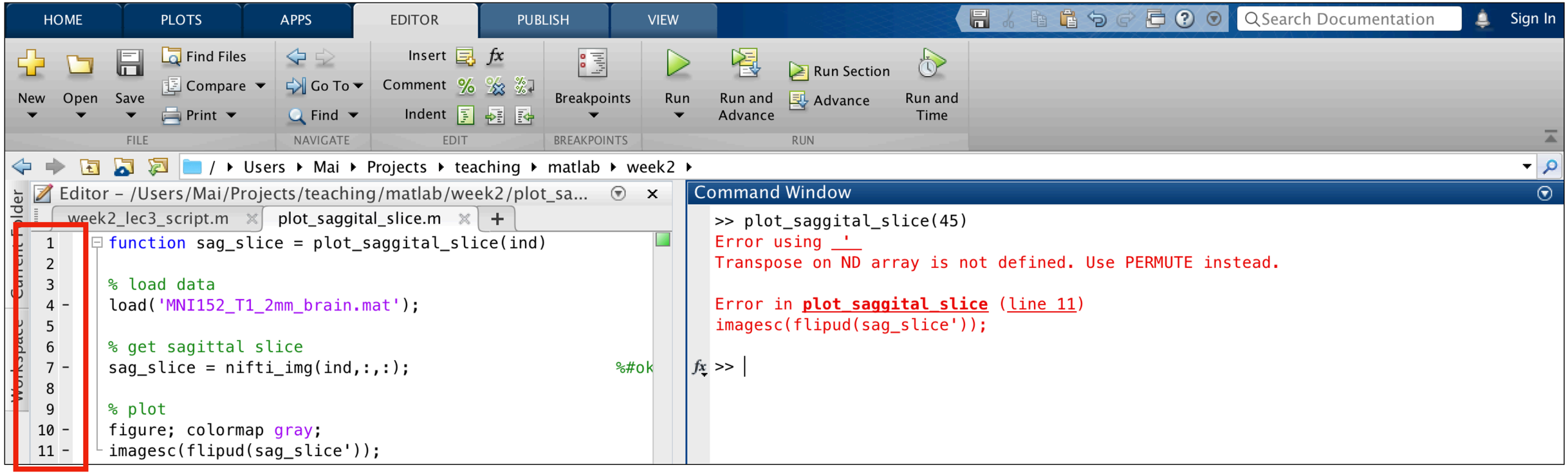
```
>> my_slice = plot_saggital_slice(45);
Error using load
Unable to read file 'MNI152_T1_2mm_brain.mat!@!@'. No such file or
directory.

Error in plot_saggital_slice (line 18)
load('MNI152_T1_2mm_brain.mat!@!@');
%#ok<*LOAD>

fx >>
```

- Describes error
- Gives function and line with error - click line number to jump to it
- Prints the line with the error

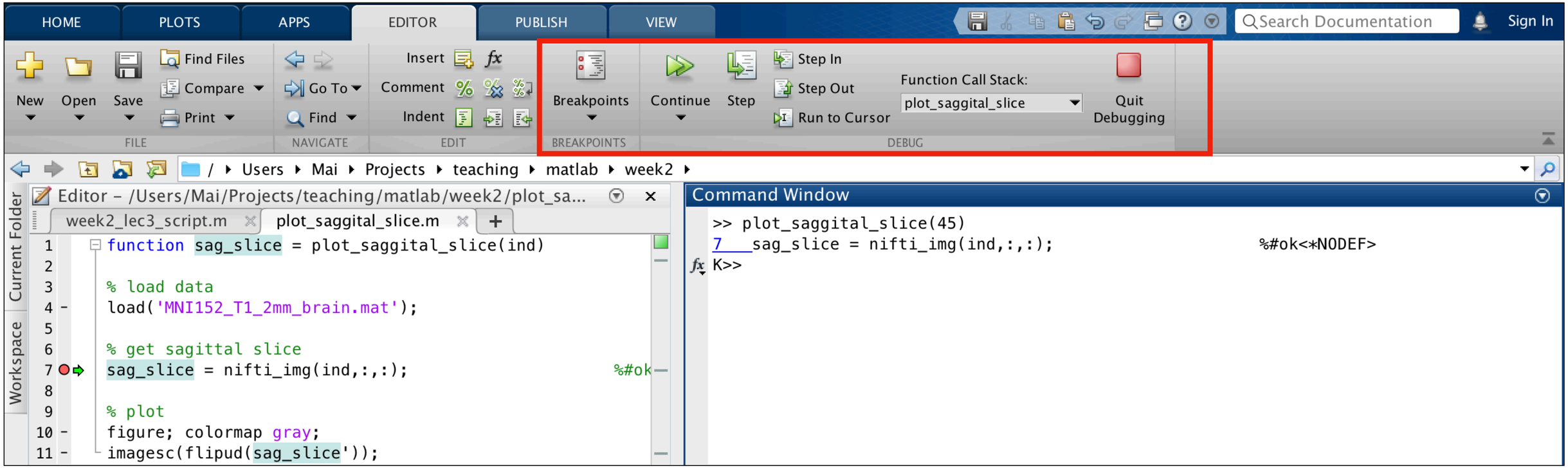
Use breakpoints



click here to make breakpoint

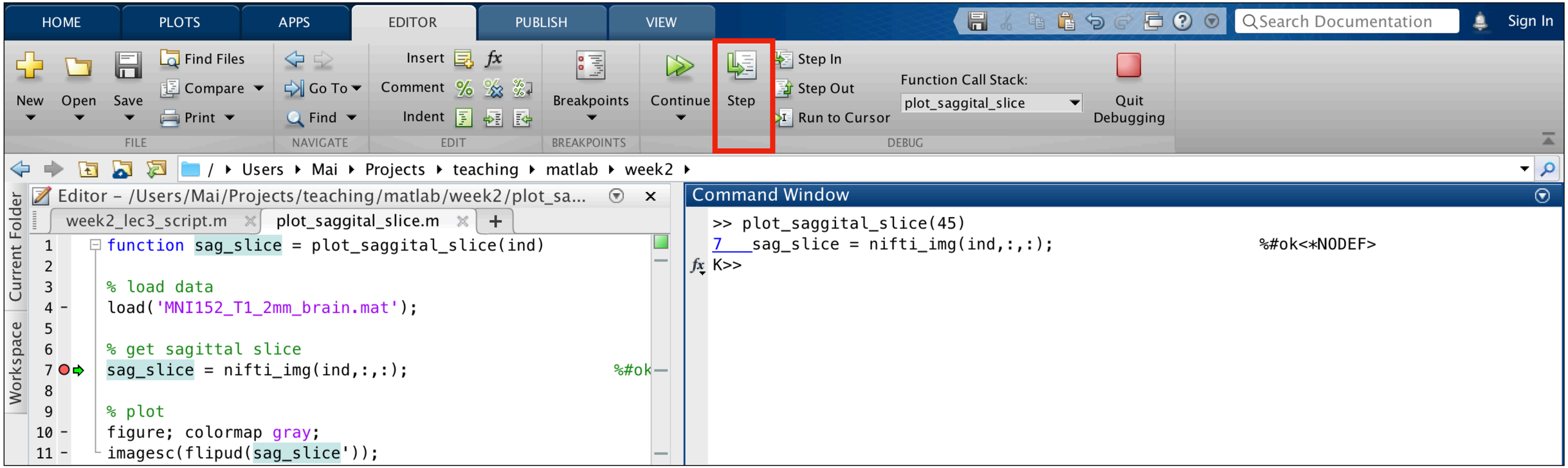
- Discovered error, but not sure what the problem is
- Insert a break point before error by clicking on line number in editor

Use breakpoints



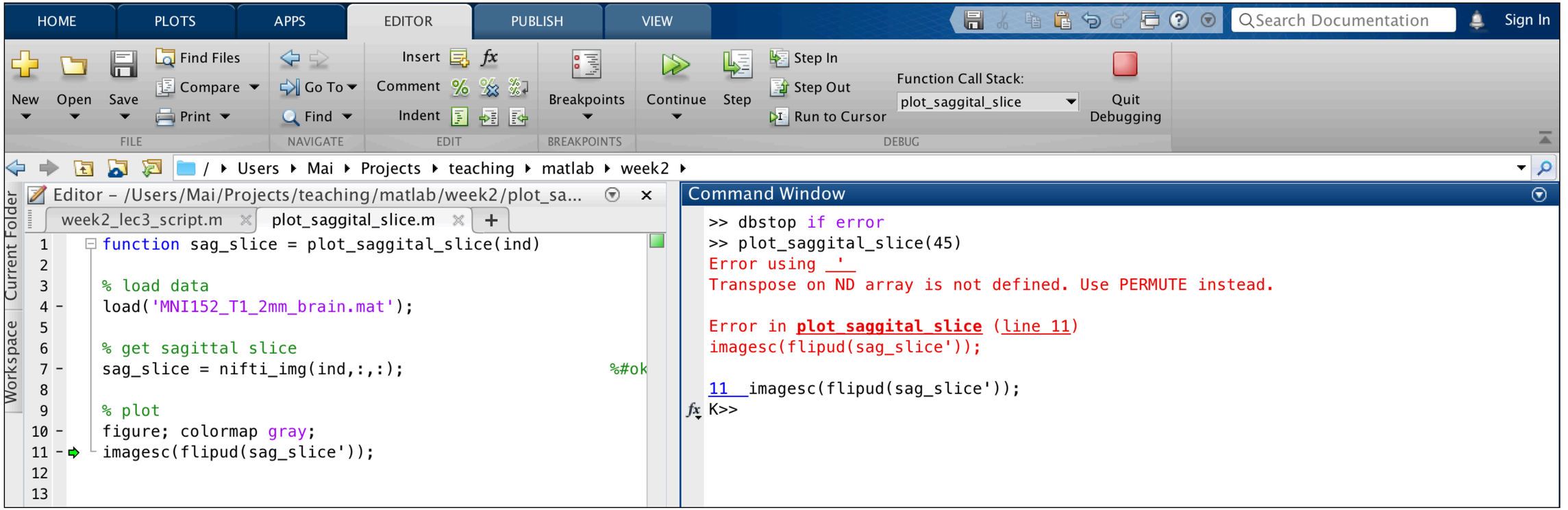
- Halts evaluating code at the breakpoint and enters debug mode
- Red circle = break point
- Green array = current line
- K>> = indicates in debut mode

Use breakpoints



- Use Step button and command window to evaluate problem

Automatically enter debug mode



Review

What are functions?
When to use
Best practices
Functions vs scripts
Scope
Modularity

Writing functions
keywords function, end
arguments
inputs
documentation

Debugging
using error messages
breakpoints
step
dbstop if error
dbquit