BIEN 203

9/18/18 Class notes

%Review

time = 1:100; %seconds

signal = sin(time);

%Generate plot

plot(time,signal)

title('title')

xlabel('x-axis label')

ylabel('y-axis label')

set(gcf,'color','w')

%Let’s obtain the figure handle (the figure’s structural array) for the generated plot.

figure\_handle = gcf; %gcf stands for “get current figure”

figure\_handle

figure\_handle.Number

figure\_handle.Units

figure\_handle.Color

figure\_handle.Children

figure

subplot(2,1,1)

subplot(2,1,2)

figure\_handle = gcf;

figure\_handle

figure\_handle.Children

figure\_handle.Children(1)

figure\_handle.Children(1).Xlabel

figure\_handle.Children(1).Title

figure\_handle.Children(1).Title.String

clc

load('T1\_sequence\_mask.mat')

%join strings together

['this' ' is ' 'important']

subject\_ID = ['S001' 'S002']

subject\_ID = ['S001';'S002']

subject\_ID = ['S001']

%This can be used to generate directory paths

['C:\Whatever',filesep,subject\_ID]

%Below is a for loop

for i = 1:1:10 %Note you set the chosen variable (i in this case) equal to an array. The for loop will perform calculation within the loop for each value in array consecutively.

disp(i) %This displays the value of i within the command window

pause(3)

end

%Below is a for loop with a numeric array that contains numbers that are not consecutive.

for i = [78 98 89 65]

disp(i)

pause(1)

end

%Let’s make a char array

subject\_ID = ['S001' ;'S002']

%Let’s look at the size of the character array

size(subject\_ID)

%Let’look at different components

subject\_ID(1)

subject\_ID(1:4)

subject\_ID(1:9)

%Let’s now make it a cell array by using curly brackets

subject\_ID = {'S001' ;'S002'}

%Let’s look at the size of the cell array

size(subject\_ID)

%Let’s look at the components now

subject\_ID(1)

subject\_ID(2)

%We can use the cell array instead of a numeric array for the for loop

for i = 1:1:length(subject\_ID) %We can use a variable to define the last integer in the array

['C:\Whatever',filesep, char(subject\_ID(i))] %You can use it to generate multiple file paths.

pause(2)

end

clc %This clears the command window, but not the command history nor workspace.

%Let’s create an array and look at it’s different components

array = [ 324 342 234]

array(2)

array(1)

array(3)

%We can save the workspace as a .mat file. Use the drop down menu in the workspace to save…

Clear

%clear will remove the variables in the workspace

%Let’s reload our workspace that we just saved.

load('workspace.mat')

%Let’s create a matrix

matrice = [ 3 42 2; 23 42 1; 6 43 87]

%Let’s look at the value on the second row and third column

matrice(2,3)

%Let’s display the matrix as an image

imagesc(matrice)

load(' T1\_sequence\_mask')

%Let’s look at the 43rd image slice of the brain mask

Answer\_D = T1\_sequence\_mask(:,:,43);

%Let’s view the image

imshow(Answer\_D)

%Let’s find the max value

max(max(Answer\_D))

%Let’s look at the unique values in the image

unique(Answer\_D)

%Let’s find the sum of all the pixels in the image

sum(((Answer\_D)))

%Load the T1\_sequence

load('T1\_sequence.mat')

%Let’s convert the T1\_seqence\_mask from 0 and 255 to ones and zeros (binary).

T1\_sequence\_mask = T1\_sequence\_mask/255;

unique(T1\_sequence\_mask) %Let’s check to make sure it’s zeros and ones

%Let’s convert from 8 bit format to double

T1\_sequence\_mask = double(T1\_sequence\_mask);

%Let’s multiple the two images to only keep the content within the brain mask

new = T1\_sequence\_mask.\*T1\_sequence;

%Let’s look at the 43rd image slice now

imagesc(new(:,:,43))

%Let’s look at the 43rd image slice from before

imagesc(T1\_sequence(:,:,43))

%Let’s look at the 43rd image slice with changes again

imagesc(new(:,:,43))

%Let’s look at other image slices from before

imagesc(T1\_sequence(:,:,107))

imagesc(T1\_sequence(:,:,80))

%Let’s look at the other image slices with changes again

imagesc(new(:,:,80))

imagesc(new(:,:,107))

%Let’s look at the dimensions of the “new” matrix at a particular y-slice

size(new(:,60,:))

%The dimensions are 255 by 1 by 150

%Let’s squeeze the dimensions

size(squeeze(new(:,60,:)))

%Now it is 255 by 150

%Let’s look at the image

imagesc(squeeze(new(:,60,:)))

%Oh no! It doesn’t look right. Let’s transpose it!

imagesc(squeeze(new(:,60,:))')

%Yay! Much better!

%Let’s look at the dimensions of the “new” matrix

size(new)

%Below is how we can obtain the size of a particular dimension

size(new,1)

size(new,3)

%Let’s create 3 orthogonal slices.

slice(new,round(size(new,1)/2),round(size(new,2)/2),round(size(new,3)/2))

%Let’s create a figure handle of the current figure and name it “lol”

lol=gcf;

%Let’s look at the children.

lol.Children

%Now the second child

lol.Children(2)

%Now the second child’s child

lol.Children(2).Children

%Now we see the 3 patches representing each of the three orthogonal slices.

%Let’s turn the visibility “off” for the second child’s first child, which is one of the planes.

lol.Children(2).Children(1).Visible = 'off'

%Let’s turn it back on

lol.Children(2).Children(1).Visible = 'on'

%Let’s get rid of the black edge colors that is preventing us from seeing the images on the slices clearly.

lol.Children(2).Children(1).EdgeColor = 'none'

lol.Children(2).Children(2).EdgeColor = 'none'

lol.Children(2).Children(3).EdgeColor = 'none';

%Wow! That is one amazing brain!

%Let’s change the colormap to gray!

colormap(gray)

%hold on, Let’s look at it 3D!!!

hold on %Don’t forget the hold on!

%use the isosurface command!

isosurface(T1\_sequence\_mask)

%Wow! That brain is brilliant!

%Let’s make another one!

figure

slice(new,round(size(new,1)/2),round(size(new,2)/2),round(size(new,3)/2))

lol=gcf;

lol.Children(1).Children(3).EdgeColor = 'none';

lol.Children(1).Children(2).EdgeColor = 'none';

lol.Children(1).Children(1).EdgeColor = 'none';

%This time let’s make the colormap “jet”

colormap(jet)

%Let’s add a colorbar too!

colorbar

%Let’s make it so that only the brain shows up in the plane and the background becomes transparent!

%Make the FaceAlpha equal to “flat”

lol.Children(3).Children(3).FaceAlpha = 'flat';

%Now how do I do this again? Was it Alpha data equal to the binary mask?

lol.Children(3).Children(3).AlphaData = T1\_sequence\_mask;

%Something go wrong? Use Google, mathworks, and help to solve this problem and get bonus points!

%generates 3D isosurface

isosurface(T1\_sequence\_mask);

%Get figure handle

lol=gca;

lol.Children(2).FaceColor = 'y';

lol.Children(2).FaceAlpha = 0.25;

xlabel('x')

ylabel('y')

zlabel('z')

set(gcf,'color','w')

axis off

view(42,14)

hold on

scatter3([23 54 100],[45, 34, 150],[34 85 100],'r','filled')

clc

%You can manually move the orientation of the figure with the mouse.

%then you can type the follow to obtain the azimuth and elevation angle.

[az el] = view

%You can manually change the orientation and type the following to bring it back to the previous view.

view(az,el)

%Below will create a random 2D matrix with values ranging from 0 to 1.

variable = rand(123);

%Let’s make a random vector array

variable = rand(123,1);

%Let’s make a for loop that goes through our new vector array

for boring = 1:1:length(variable)

variable(boring)

pause(1)

end

%We are going to define this value to be used in an if condition.

Special = variable(5)

%Let’s add an if condition

for boring = 1:1:length(variable)

disp(num2str(boring))

if variable(boring) == Special

disp([num2str(boring) ' THIS ONE IS SPECIAL!!'])

end

pause(1)

end

%Let’s add an else clause!

for boring = 1:1:length(variable)

% disp(num2str(boring))

if variable(boring) == Special

disp([num2str(boring) ' THIS ONE IS SPECIAL!!'])

else

disp(['Sorry number ' num2str(boring) ', you are not special :('])

end

pause(1)

end

%Let’s graph a 2D scatter plot and overlay a line plot on top.

figure

subplot(2,1,1)

scatter(time,signal)

plot(time,signal)

%... oh no! the plot overwrote the scatter plot in the figure

%Be sure to use hold on!

figure

subplot(2,1,1)

scatter(time,signal)

hold on

plot(time,signal)

subplot(2,1,2)

time = 1:.1:100; %seconds

signal = sin(time);

scatter(time,signal)

hold on

plot(time,signal)

clc

**Class exercise:**

%Load T1\_sequence.mat

load('T1\_sequence\_mask.mat');

%a) Find the dimensions of the T1\_sequence\_mask matrix and save it under the variable name "T1\_dimensions".

T1\_dimensions = size(T1\_sequence\_mask);

%b) Use the length command to find the length of the largest dimension of the T1\_sequence\_mask matrix and save it under the variable name "T1\_length".

T1\_length = length(T1\_sequence\_mask);

%c) Find the value in the T1\_sequence\_mask matrix at the coordinates (34,86,43) and save it under the variable name "Answer\_C".

Answer\_C = T1\_sequence\_mask(34,86,43);

%d) Find the values in the x-y plane of the T1\_sequence\_mask matrix at the z-slice of 43 and save it under the variable name "Answer\_D".

% Assume the dimensions of the T1\_sequence\_mask matrix are listed consecutively as x, y, and z.

Answer\_D = T1\_sequence\_mask(:,:,43);

%Display the x-y plane of T1\_sequence\_mask Matrix at the z-slice that is half way through the stack.

imagesc(T1\_sequence\_mask(:,:,size(T1\_sequence\_mask,3)/2));

%Display 3 orthogonal planes (x-y plane, x-z plane, and y-z plane

figure\_handle = slice(double(T1\_sequence\_mask),size(T1\_sequence\_mask,1)/2,size(T1\_sequence\_mask,2)/2,size(T1\_sequence\_mask,3)/2);

%Remove the EdgeColor by setting it to 'none' for each plane

figure\_handle(1).EdgeColor = 'none';

figure\_handle(2).EdgeColor = 'none';

figure\_handle(3).EdgeColor = 'none';

%Change the colormap to gray

colormap(gray)

%isosurface

isosurface(T1\_sequence\_mask)