## Repeating With Loops

## Learning Objectives

* Explain what a for loop does.
* Correctly write for loops that repeat simple commands.
* Trace changes to a loop variable as the loops runs.
* Use a for loop to process multiple files

**Part – Intro to Loops**

Why we need loops

What we would do without loops:

Clearing the workspace

Sum function

% clear workspace

clear all

clc

% read in data

cd('Data')

patient\_data=csvread('inflammation-01.csv');

daysum=sum(patient\_data(:,1));

disp(['sum on day 1: ' num2str(daysum)]);

daysum=sum(patient\_data(:,2));

disp(['sum on day 2: ' num2str(daysum)]);

Using variables as placeholders (replace number with variable)

% read in data

cd('Data')

patient\_data=csvread('inflammation-01.csv');

day=39; % variable containing day number

daysum=sum(patient\_data(:,day)); % sum over all patients

disp(['sum on day ' num2str(day) ': ' num2str(daysum)]); % display sum

Turning it into a loop (just write syntax)

% read in data

clear all

clc

cd('Data')

patient\_data=csvread('inflammation-01.csv');

for day=1:40 % variable containing day number

daysum=sum(patient\_data(:,day)); % sum over all patients

disp(['sum on day ' num2str(day) ': ' num2str(daysum)]);% display sum

end

Explain syntax

Set breakpoint and discuss

Step through (F10)

Do every second day (explain syntax 1:2:40)

Replace 40 with size function

for day=1:2:size(patient\_data,2) % variable containing day number

daysum=sum(patient\_data(:,day)); % sum over all patients

disp(['sum on day ' num2str(day) ': ' num2str(daysum)]); % display sum

end

CHALLENGE:

% Using the worked example as a guide, write a script that will calculate

% the sum of the inflammation of each patient over 40 days for the file

% inflammation-01 starting from day 40 to day 1. Display the sum for each

% day to the command line.

% Change your loop so that it only displays the sums on days 4,7,13,16,28

% and 33.

EXTENSION:

%% Extension

% Modify your loop to instead PLOT the inflammation data of ALL patients on days 4,7,13,16,28 and 33

% Use subplots with two rows and three columns

% Make sure each of your subplot has a title that indicates the day.

% Save this figure as a png file to disk.

**Part 2 – Using A Loop To Analyse Data**

Let’s write a loop that tries to print the names of each one of our files:

Write script and run

% loop to print names of data files

for idx = 1:4

file\_name = sprintf('inflammation-%d.csv', idx);

disp(file\_name);

end

Explain sprint:

“The sprintf function is useful when we want to generate MATLAB strings based on a template. In our case, that template is the string inflammation-%d.csv. sprintf generates a new string from our template by replacing the %d with the data referred to by our second argument, idx.”

Trace execution of loop

Filename that’s printed is wrong

Modify template:

“We’ve replaced %d in our earlier template with %02d. With this, we’re specifying that we want our data to be displayed with a minimum width of 2 characters, and that we want to pad with 0 for data that isn’t at least 2 digits long.”

for idx = 1:4

file\_name = sprintf('inflammation-%02d.csv', idx);

disp(file\_name);

end

Use this to loop through data files in analyse.m

Make file\_name and img\_name variables, change image saving function

file\_name = sprintf('inflammation-%02d.csv', idx);

img\_name = sprintf('patient\_data-%02d', idx);

CHALLENGE:

% Insert for loop so that the script loads in each data file sequentially,

% and saves out a heatmap for each data file (using imagesc).

% HINT: loop using the variable idx.

**Part 3 – Debugging For Loops**

Make an error in the loop, debug it

CHALLENGE:

* Pie.