Using ImageJ for Manual and Automated Image Analysis

Jeff LeDue and Claire Bomkamp

November 4 2016

Overview of today's session

- Introduce ImageJ and how to manually manipulate images
- Discuss different options for automation in ImageJ (ie macro recorder, batch processing, macros in various langauges)
- Go through an example of the macro recorder
- Extend macro recorder example to process all files in a folder
- Attempt example from Chris as a group

- Goals:
  - Familiarity with the different options for automation in ImageJ
  - “Hands-on” experience using the macro recorder and a few key commands to batch process

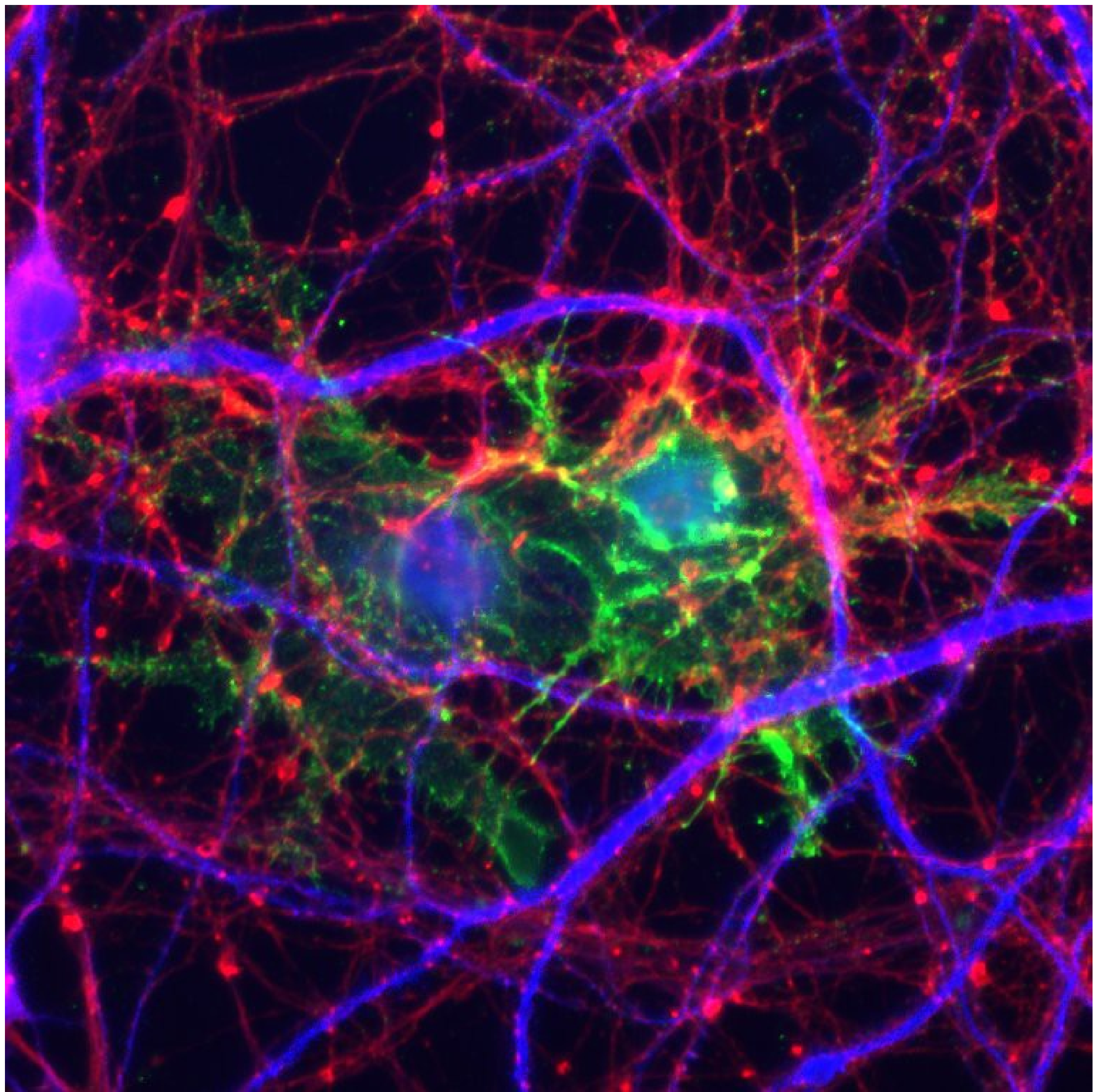
ImageJ vs. Other Things

- Good stuff:
  - Commonly used, time-tested
  - Open source, lots of plugins available
  - Understands most image formats you might want to use
  - Support for scripting in multiple languages (Fiji)
  - Easy to move between manual and automated analysis
- Bad stuff:
  - Jython is not Python, etc.
  - Scripting stuff is tacked on top of a graphical interface → performance issues
  - Occasional really weird bugs

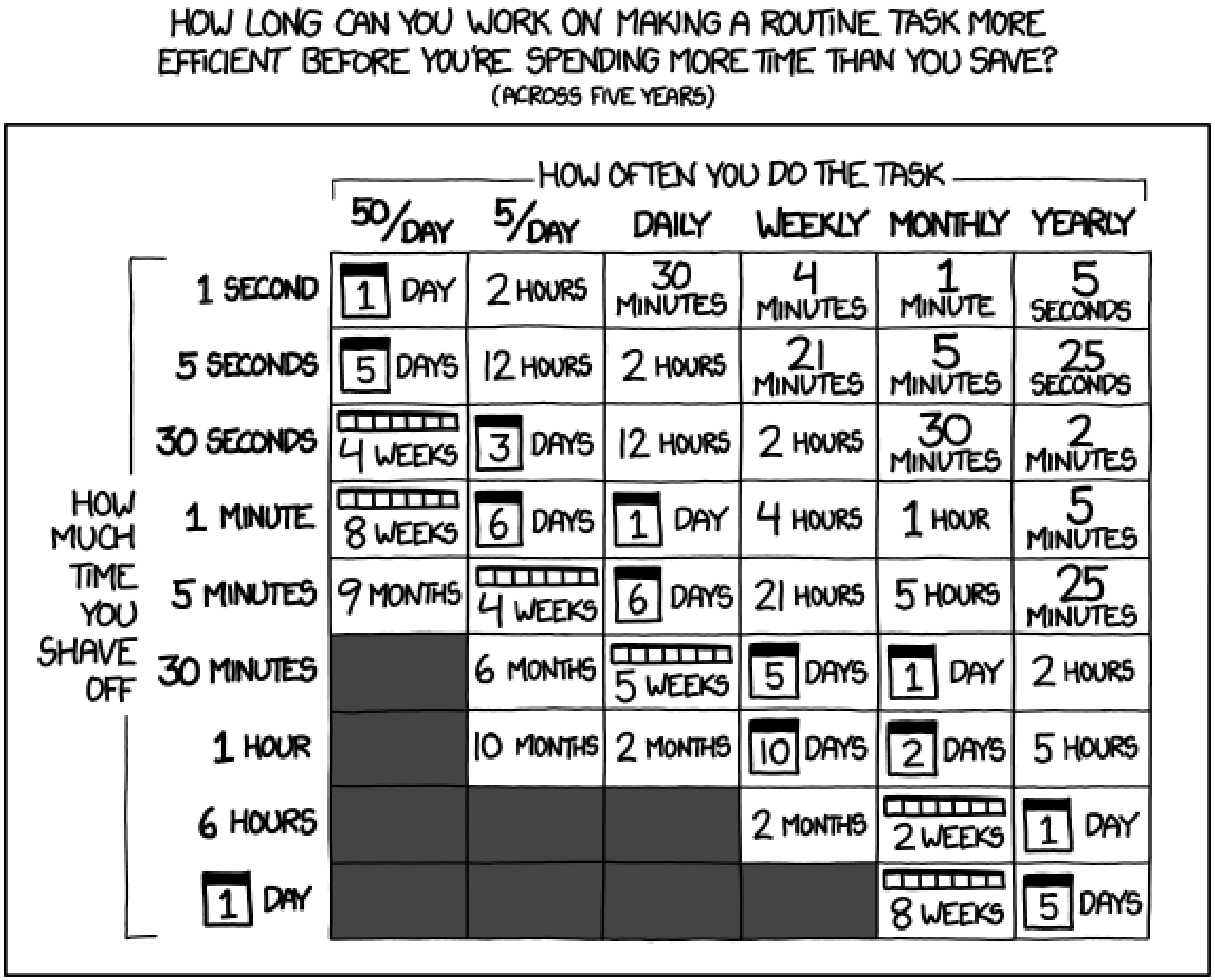
https://www.pinterest.com/pin/367817494548456209/

ImageJ Basics

- Good to know how to run your analysis by hand, for troubleshooting purposes and to check that everything is working correctly!
- Example from Claire's work.



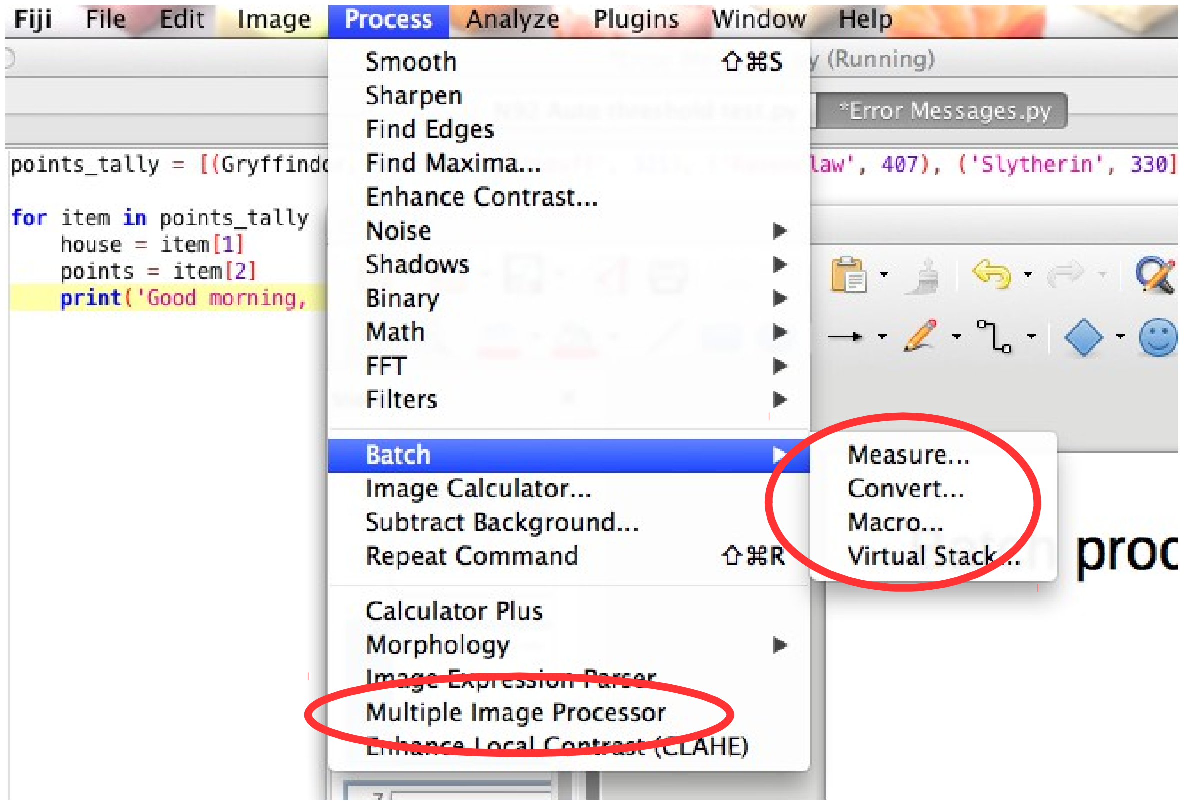
Automation



HOW LONG CAN YOU WORK ON MAKING A ROUTINE TASK MORE EFFICIENT BEFORE YOU'RE SPENDING MORE TIME THAN YOU SAVE?
(ACROSS FIVE YEARS)

		HOW OFTEN YOU DO THE TASK					
		50/DAY	5/DAY	DAILY	WEEKLY	MONTHLY	YEARLY
HOW MUCH TIME YOU SHAVE OFF	1 SECOND	1 DAY	2 HOURS	30 MINUTES	4 MINUTES	1 MINUTE	5 SECONDS
	5 SECONDS	5 DAYS	12 HOURS	2 HOURS	21 MINUTES	5 MINUTES	25 SECONDS
	30 SECONDS	4 WEEKS	3 DAYS	12 HOURS	2 HOURS	30 MINUTES	2 MINUTES
	1 MINUTE	8 WEEKS	6 DAYS	1 DAY	4 HOURS	1 HOUR	5 MINUTES
	5 MINUTES	9 MONTHS	4 WEEKS	6 DAYS	21 HOURS	5 HOURS	25 MINUTES
	30 MINUTES	▨	6 MONTHS	5 WEEKS	5 DAYS	1 DAY	2 HOURS
	1 HOUR	▨	10 MONTHS	2 MONTHS	10 DAYS	2 DAYS	5 HOURS
	6 HOURS	▨	▨	▨	2 MONTHS	2 WEEKS	1 DAY
	1 DAY	▨	▨	▨	▨	8 WEEKS	5 DAYS

Batch processing for simple tasks



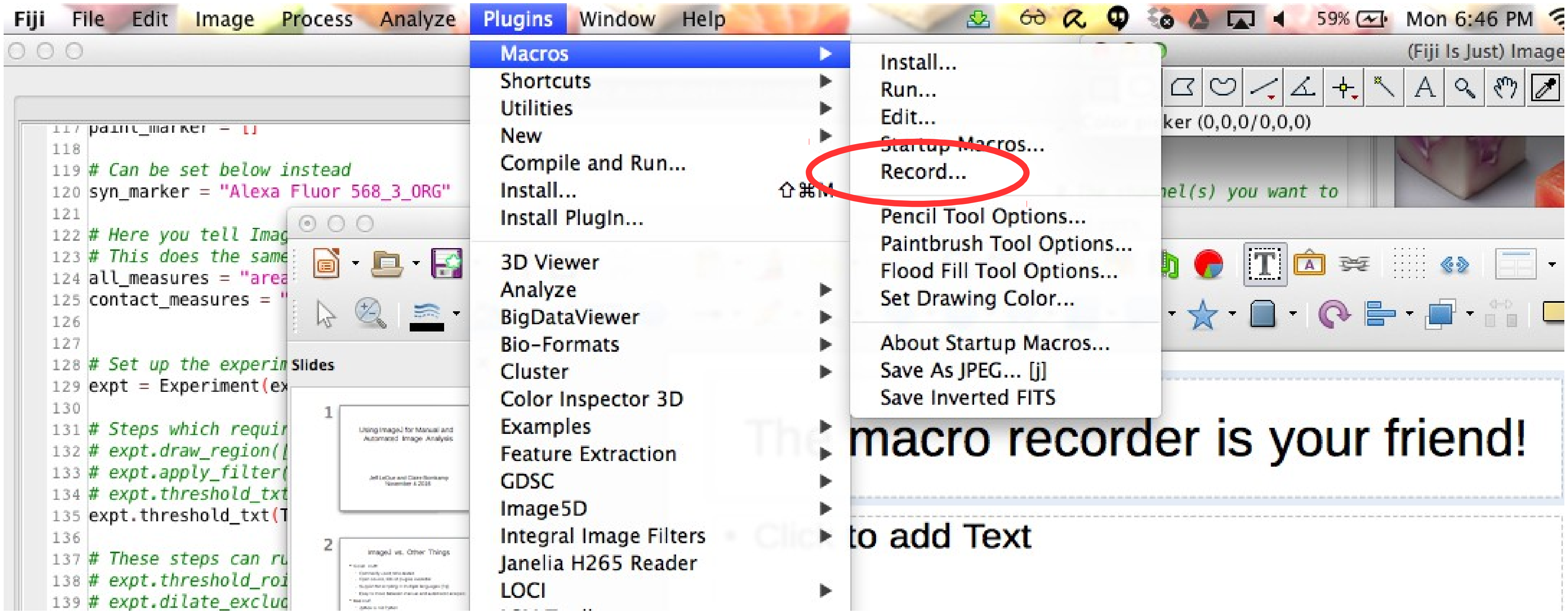
```
points_tally = [(Gryffindo                              law', 407), ('Slytherin', 330]

for item in points_tally
    house = item[1]
    points = item[2]
    print('Good morning,
```

What language should I use?

- If you don’t want to deal with one language pretending to be another, use either Java or ImageJ’s Macro language
- Or, use whichever language you will wish you had started learning now, two years from now
- http://imagej.net/Jython_Scripting

The macro recorder is your friend!



```
# Can be set below instead
syn_marker = "Alexa Fluor 568_3_ORG"

# Here you tell Imag
# This does the same
all_measures = "area
contact_measures = "


# Set up the experim
expt = Experiment(ex

# Steps which requir
# expt.draw_region(
# expt.apply_filter(
# expt.threshold_txt
expt.threshold_txt(T

# These steps can ru
# expt.threshold_roi
# expt.dilate_exclud
```

Examples

- First: use macro recorder to process a single file
- Extend the macro recorder result to batch process all files in a folder
- Claire will show the equivalent in Python

ImageJ macro recorder example

- This example uses data from one of the auto-head fixing cages from Tim’s lab. (Show paper, provide some context)
- The cages generate potentially 100’s of 256x256x~930 frame XYT stacks per day.
- We need a quick way to do quality control, to weed out files where perhaps someone was adjusting the illumination or other mishaps.
- By calculating delta F over Fo we could see easily when a particular file was corrupted (changes in illumination gave signals way to large to have originated from GCaMP)
- We will try to build up a macro that process all the recordings in a folder starting with the macro recorder to record all the steps.

Follow the steps 1/2:

- Step one, open Fiji.
- Click Plugins>Macros>Record…
- Choose either “Macro” or “Java”
  - Translating to Python (and probably other languages) is easier from Java
- Go to file>open and navigate to the tif file in the single tiff folder of your example files folder
- Go to image>rename, type in current and accept.
- Do image>scale and type in 0.5 for XY
- Select the window with the original and close
- Select the current-1
- Do image>stacks>z project, pick average intensity
- Do process image calculator, subtract AVG_current-1 from current-1

Follow the steps 2/2:

- Do process>image calculator>divide Result of current-1 by AVG_current-1, select the option for 32 bit float result.
- In turn select Result of current-1, AVG_current-1, current-1 and close them without saving
- Do image>adjust>Brightness & Contrast, click set and type -0.02 to .2, ie scale the gray values from -2 to 20% changes.
- Do image>look up tables>Aselfmade3
- When you are done, name your macro and then click create.
- At this point if you replace the full path and filename in the first line you can repeat these steps on any file!

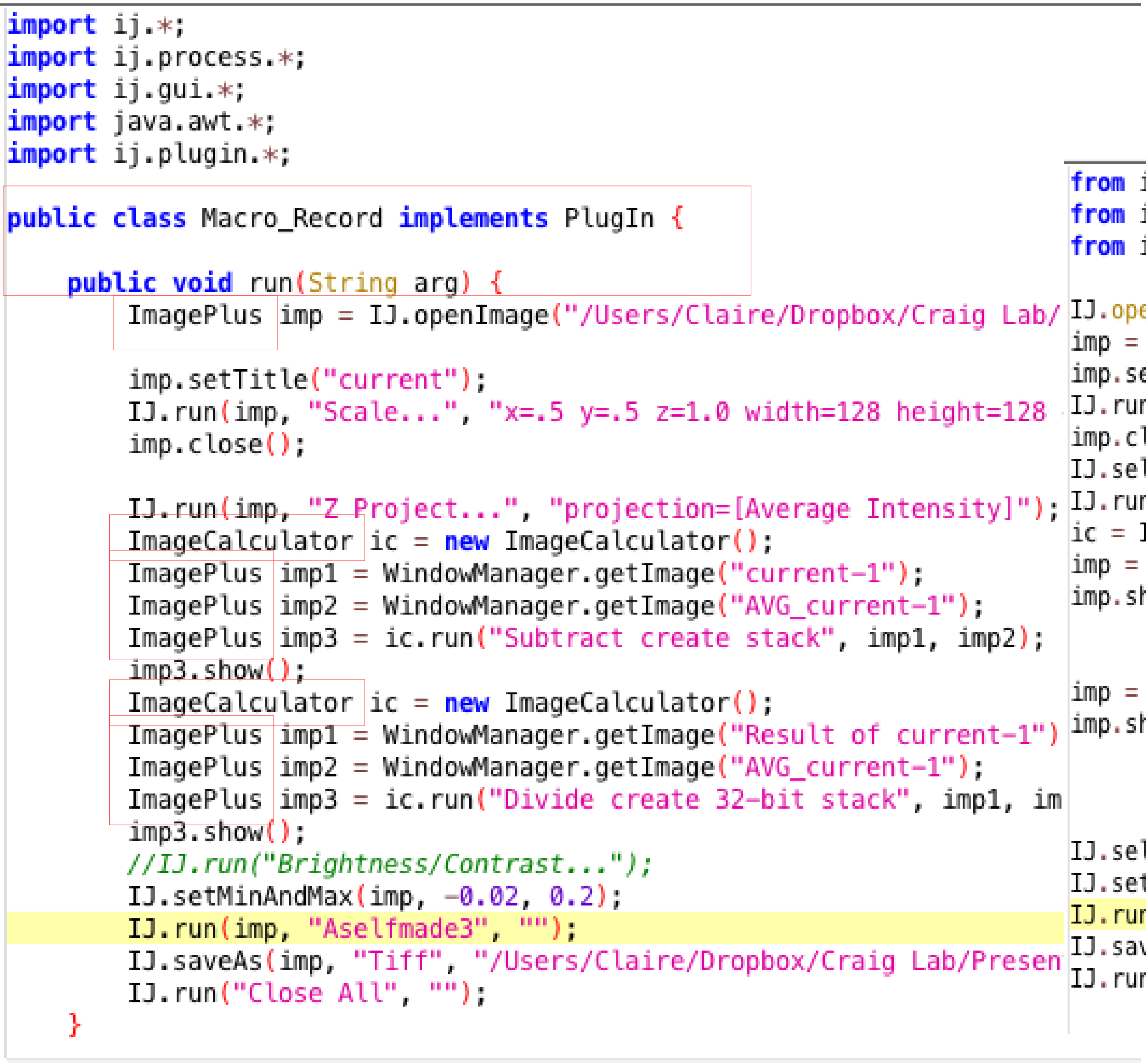
Macro recorder → macro 1/2:

- I like to use the macro recorder to give me the “guts” of the processing that is needed.
- We can then place the “guts” inside a loop which will apply our processing steps to each file in a folder for example.
- To do this we need a couple of commands.
- getDirectory & getFileList
- And a for loop.
- Create a new macro file in the macro editor and save it with the same name as your macro recorder result with _LOOPED at the end (or similar)

Macro recorder → macro 2/2:

- At the top add:
- path=getDirectory("Choose a Directory");
- ls = getFileList(path);
- The first one is used to pick the folder with the files in it.
- The second one gives you a list of the files.
- Then add the basic structure of a loop:
  - for (i=0; i<ls.length; i++)
  - {
  - }
- The “guts” go between the curly braces. Copy them in from your macro recorder result.
- Two further changes are needed, make your full path and file name out of the path and file list variables
  - fn=path+ls[i];
  - Do this just before your open command, and then replace the full path and file with fn
- And rename your image back to the original file name after the operations
  - rename(ls[i]);
- Try it on the 5 files tiff folder.

Translating a Java macro to Python



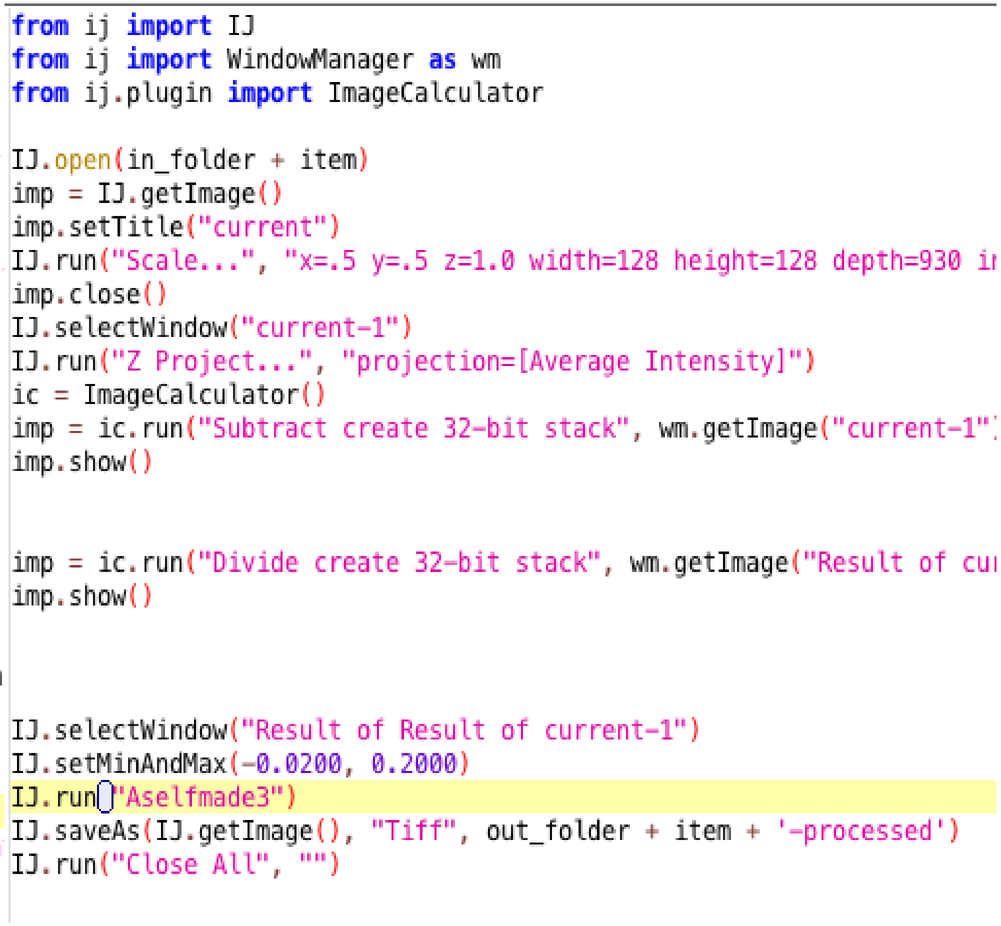
```
import ij.*;
import ij.process.*;
import ij.gui.*;
import java.awt.*;
import ij.plugin.*;

public class Macro_Record implements PlugIn {

    public void run(String arg) {
        ImagePlus imp = IJ.openImage("/Users/Claire/Dropbox/Craig Lab/

        imp.setTitle("current");
        IJ.run(imp, "Scale...", "x=.5 y=.5 z=1.0 width=128 height=128
        imp.close();

        IJ.run(imp, "Z Project...", "projection=[Average Intensity]");
        ImageCalculator ic = new ImageCalculator();
        ImagePlus imp1 = WindowManager.getImage("current-1");
        ImagePlus imp2 = WindowManager.getImage("AVG_current-1");
        ImagePlus imp3 = ic.run("Subtract create stack", imp1, imp2);
        imp3.show();
        ImageCalculator ic = new ImageCalculator();
        ImagePlus imp1 = WindowManager.getImage("Result of current-1")
        ImagePlus imp2 = WindowManager.getImage("AVG_current-1");
        ImagePlus imp3 = ic.run("Divide create 32-bit stack", imp1, im
        imp3.show();
        //IJ.run("Brightness/Contrast...");
        IJ.setMinAndMax(imp, -0.02, 0.2);
        IJ.run(imp, "Aselfmade3", "");
        IJ.saveAs(imp, "Tiff", "/Users/Claire/Dropbox/Craig Lab/Presen
        IJ.run("Close All", "");
    }
```



```
from ij import IJ
from ij import WindowManager as wm
from ij.plugin import ImageCalculator

IJ.open(in_folder + item)
imp = IJ.getImage()
imp.setTitle("current")
IJ.run("Scale...", "x=.5 y=.5 z=1.0 width=128 height=128 depth=930 in
imp.close()
IJ.selectWindow("current-1")
IJ.run("Z Project...", "projection=[Average Intensity]")
ic = ImageCalculator()
imp = ic.run("Subtract create 32-bit stack", wm.getImage("current-1"
imp.show()


imp = ic.run("Divide create 32-bit stack", wm.getImage("Result of cu
imp.show()



IJ.selectWindow("Result of Result of current-1")
IJ.setMinAndMax(-0.0200, 0.2000)
IJ.run("Aselfmade3")
IJ.saveAs(IJ.getImage(), "Tiff", out_folder + item + '-processed')
IJ.run("Close All", "")
```

Translating a Java macro to Python

- Delete semicolons, brackets, unnecessary indents (Find & replace is your friend)
- Delete object type things
- Delete Java nonsense
- Run it and see what breaks!

- One thing that will definitely break: Imports! Find the name of the broken thing on https://imagej.nih.gov/ij/developer/source/

Chris's example

- Chris to introduce describe the data and what is needed.
- We will try and build a macro together.

Batch Mode

- Your analysis will probably run faster if ImageJ doesn’t have to spend time displaying each image on the screen!

```
setBatchMode(true)
setBatchMode(false)
```

- However, due to ImageJ being ImageJ, the ROI Manager tends to not work in batch mode.

Troubleshooting

- Familiarize yourself with common error messages in your chosen language
- If something isn’t working and you think it should be, try doing it by hand
- Add lots and lots of print statements and/or pauses in the macro so you can see what’s going on

Common error messages (Python)

- SyntaxError: ("mismatched input '\\n' expecting COLON"
  - Probably a punctuation mark missing or added
- SyntaxError: ("no viable alternative at input ']'
  - Most likely missing closing parentheses
- TypeError: cannot concatenate 'str' and 'int' objects
  - Change numbers to look like this: str(n)
- NameError: name 'Gryffindor' is not defined
  - Put quotes around it! Otherwise it’s treated as a variable name

Problems when translating from the Java recorder to Python

```
IJ.run(imp, "Mexican Hat Filter",
"radius=5")
```

- NameError: name 'IJ' is not defined
  - IJ is a module (one of several) containing ImageJ-specific commands. You have to import it before you can use it:
  - `from ij import IJ`
- NameError: name 'imp' is not defined
  - ‘imp’ is short for ImagePlus, and refers to the current image. However, Jython doesn’t know that. Here’s how to fix it:
  - `Imp = IJ.getImage()`

A useful thing!

- If you have an unfortunate life like mine where your analysis can’t be 100% automated, this is something you need to know!

```
ij.gui.WaitForUserDialog("Human,
please help me!").show()
```

- You may also need dialog boxes where you can input numbers, select options, and such. These aren’t that hard to make!