ImageJ Macro Tutorial

9/3/2015

Keria Bermudez-Hernandez

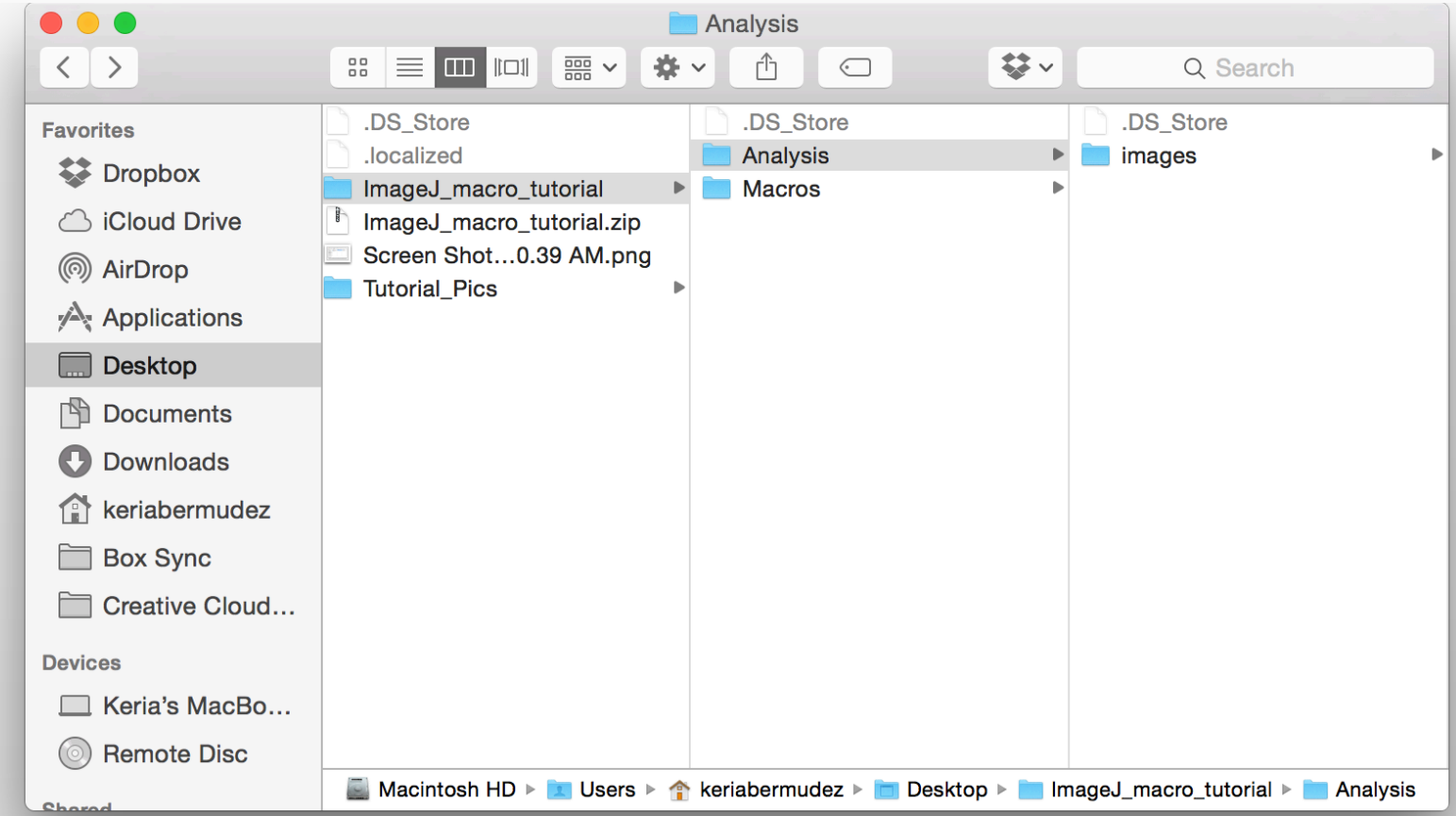
kb1139@nyumc.org

ImageJ/Fiji

- Fiji is an image processing package. It can be described as a "batteries-included" distribution of ImageJ (and ImageJ2), bundling Java, Java3D and a lot of plugins organized into a coherent menu structure.
- A macro is a simple program that automates a series of ImageJ commands.
- Download and install Fiji http://fiji.sc/Downloads#Fiji
- Download zip file an place in Desktop
- If you haven't done so. Go to

https://github.com/keriber/ImageJMacroTutorial

Folder Structure



Outline

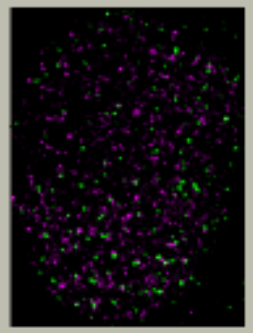
- Where are macros and how to install them
- I will show you the syntax of macro language
- And I will show you how to write a macro with an actual example

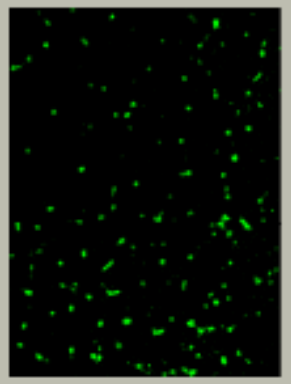
Macro Example

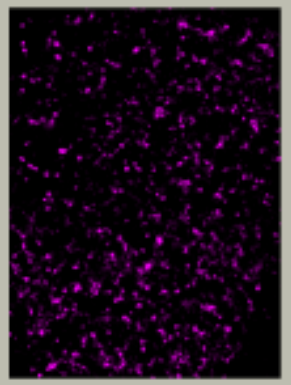
1. Separating Channels. Save each channel.

2. Segmenting Images

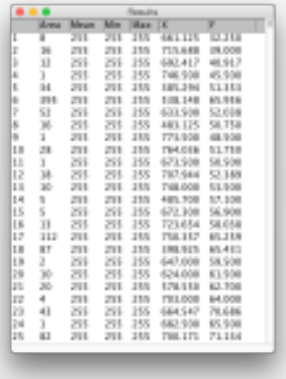
3. Find properties of clusters.
   Save all properties in a .csv file
   Save all cluster region of interests(ROIs)

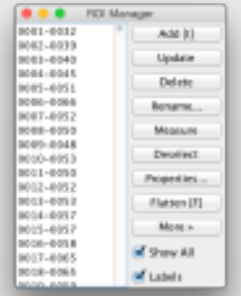






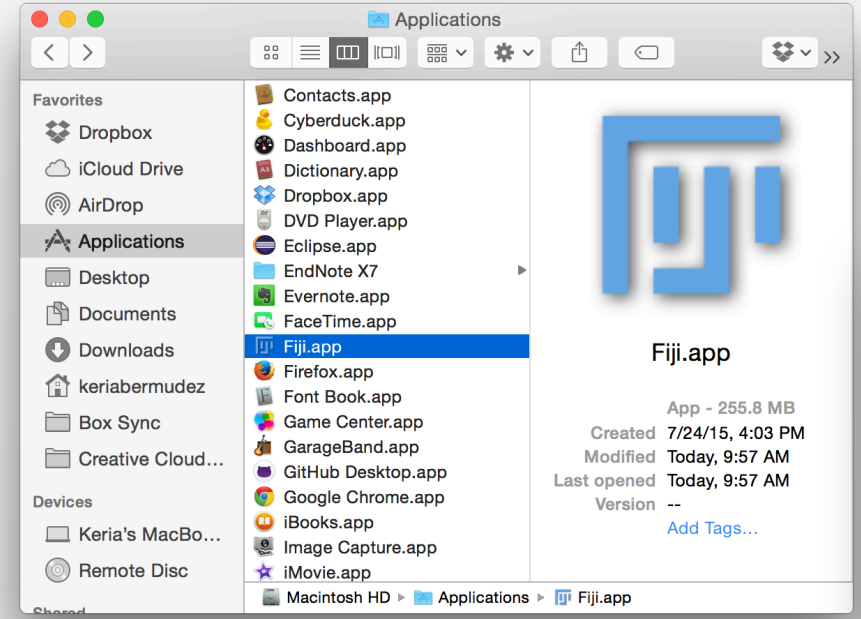
AND

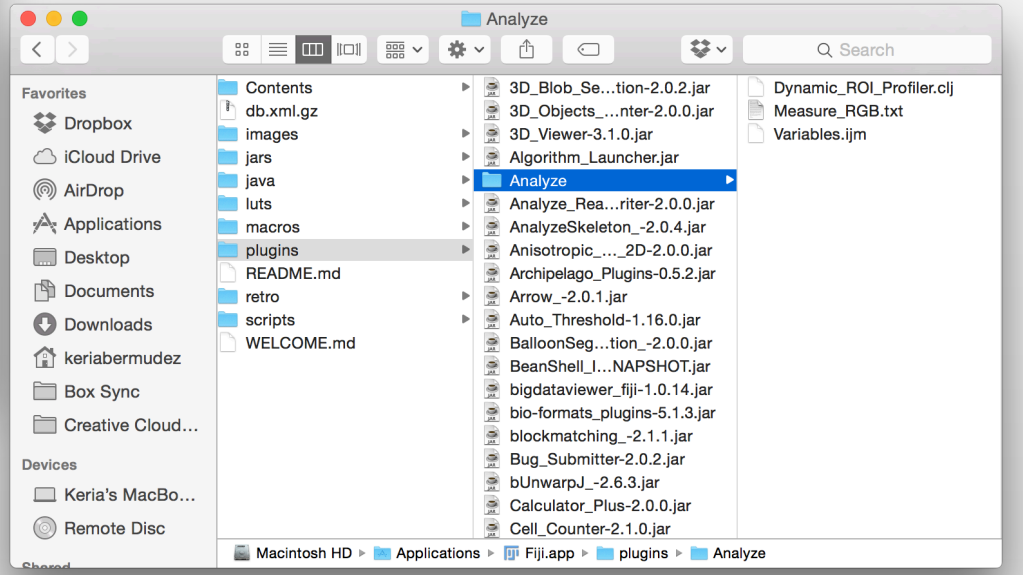




Where are macros

- Where are macros?





Lets Install our Macro

1. Rename HelloWorld.ijm to Hello_World.ijm (macros can be .txt or .ijm but must have _)
2. Copy the file to the folder:

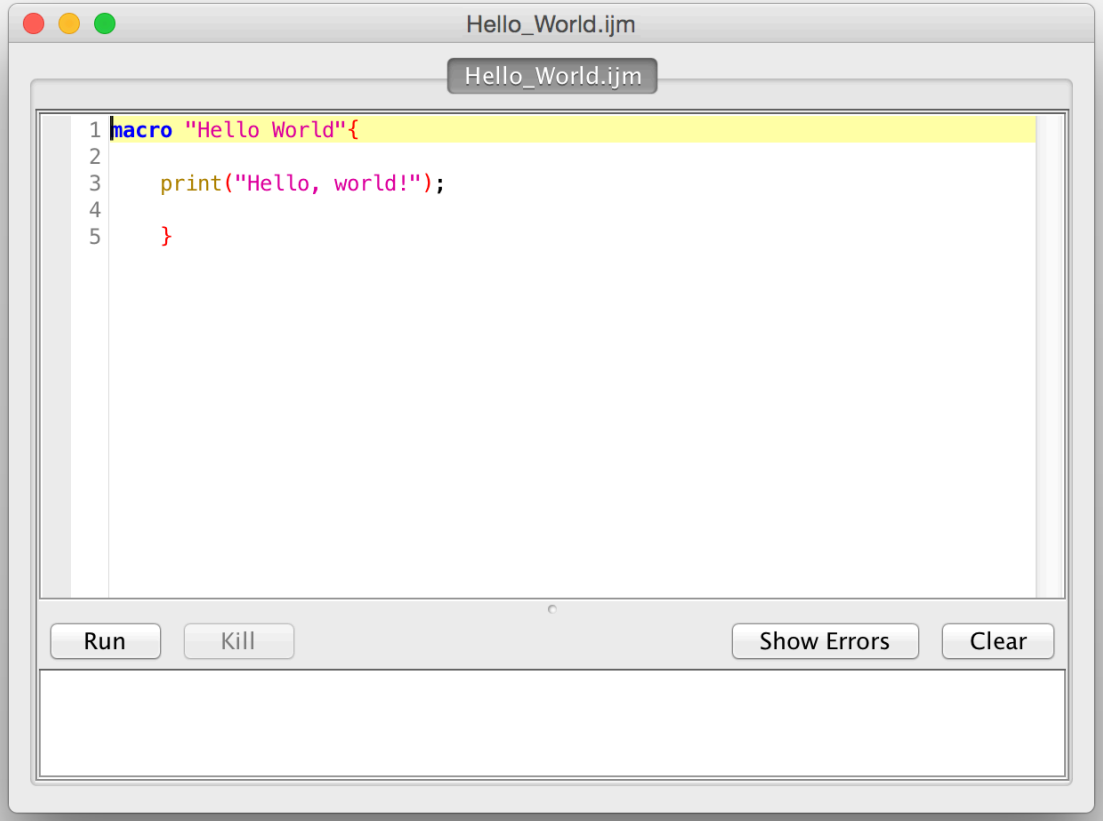
   plugins/Analyze
3. Install plugin in ImageJ by going to:

   Plugin>>Macros>>Install

   and selecting the macro

Hello_World.ijm Macro

1. Open macro by dragging file to ImageJ
2. The script Editor open
3. Click Run



ImageJ Macro Language Syntax

- Variables
  - You define a variable by giving the variable a value
  - Variables can store different data types
- Conditional Statements or If Statements
  - They test if a condition is True and execute a code
- Looping Statements or Iteration Statements
  - Three Types:
    - For
    - While
    - Do while
- Functions
  - Self contained unit of program designed to accomplish a task
  - Arguments are passed to function
  - Parameters appear in function definitions

http://rsb.info.nih.gov/ij/developer/macro/macros.html

Variables and Data types

- Open Variables.ijm
- Location:
  /Desktop/ImageJ_macro_tutorial /Macros/syntax_examples/

Variables.ijm (Running)

Variables.ijm (Running)

```
macro "Variables"{

    //1) Numbers
      v = 1.23;
      x = 2.55;
     print(v);
     print(x);

    //2) Strings
     y = "Hello World";
     print(y);

    //3) Arrays
     z = newArray(10, 20, 50);

    //Indexing Arrays
     print(z[0]);

}
```

Run | Kill | Show Errors | Clear

```
Started Variables.ijm at Thu Sep 03 13:59:08 EDT 2015
```

Conditional Statements

- Open LoopingStatements.ijm
- Location:
  /Desktop/ImageJ_macro_tutorial /Macros/syntax_examples/

ConditionalStatements.ijm

ConditionalStatements.ijm

```
macro "ConditionalStatements"{

    y = 55;

    if (y <100){
        print("The value of y is less than a 100");
        }
    else{
        print("The value of y is not less than a hundred");
        }
    }
```

Run Kill Show Errors Clear

Looping Statements

- Open LoopingStatements.ijm

Location:
/Desktop/ImageJ_macro_tutorial /Macros/syntax_examples/

Every loop has
- initiator(i=0)
- limiter(i<10)
- incrementer (i++)

i++ means i = i+1

*LoopingStatements.ijm

LoopingStatements.ijm

```
macro "Looping Statements"{

    //Looping statements are used to repeatadly run a block of code. The ImageJ macro language has three looping

    //1) for - runs a block of code a specified number of times
    print("This is an example of a for loop");
    for (i=0;i<10;i++){
        print(i);
    }

    //2)while - repeatedly runs a block of code while a condition is true
    print("This is an example of a while loop");
    i=0;
    while (i<10){
        print(i);
        i++;
    }

    //3)do...while - runs a block of code once then repeats while a condition is true
    print("This is an example of a do while loop");
    i = 0;
    do {
      print(i);
      i = i + 10;
    }
    while (i<=90);
```

Run Kill Show Errors Clear

Lets Build Our Macro

- Open the macro recorder
  Plugins>>Macros>>Record
- Open an image that is located in
  /Desktop/ImageJ_macro_tutorial /Analysis/images/
  File>>Open
- Split Channels
  Image>>Color>>Split Channels
- Convert to Binary
  Image>>Adjust>>Threshold change to Otsu and click apply
- Analyze particles
  Analyze>>Analyze Particles

Lets Build our Macro

- Open AnalyzeParticlesBatch_0.ijm and AnalyzeParticlesBatch_1.ijm
  Location:
  /Desktop/ImageJ_macro_tutorial /Macros/analyze_particles/
- I build macros by copying from the macro recorder and by finding functions in
  http://rsb.info.nih.gov/ij/developer/macro/functions.html

AnalyzeParticlesBatch_0.ijm

AnalyzeParticlesBatch_0.ijm | AnalyzeParticlesBatch_1.ijm

```
macro "AnalyzeParticlesBatch_0" {

    //1)Selecting directory of image

    //2) Generate directories to save channels, masks



    //3) Open RGB image


    //4) Get RGB image name


    //5) Split RGB into red, green, and blue channels


    //6) Select red channel and save the image in directory


    //7) Run threshold, make binary, run AnalyzeParticles command

    //8) Save results


    //9) Save ROIs
```

AnalyzeParticlesBatch_1.ijm

- Macro that runs the ImageJ comands for one image

AnalyzeParticlesBatch_1.ijm

```
macro "AnalyzeParticlesBatch_1" {

    //1)Selecting directory of image and mask

    image_path =  File.openDialog("Select a RGB image");
    dir = File.getParent(image_path);
    save_path = File.getParent(dir);

    print(image_path);
    print(dir);
    print(save_path);

    //2)Generate directories to save channels, masks

    File.makeDirectory(save_path+"//"+"Red") ;
    File.makeDirectory(save_path+"//"+"Green");
    File.makeDirectory(save_path+"//"+"ROIs") ;

    //3)Open RGB image

    open(image_path);

    //4)Get RGB image name

    image = File.getName(image_path);
    print(image);
    len = lengthOf(image);
    image_name = substring(image, 0, len-4); //substring
    print(image_name);

    //5)Split RGB into red, green, and blue channels

    run("Split Channels");

```

AnalyzeParticlesBatch_2.ijm

- Macro that runs the ImageJ commands for multiple images

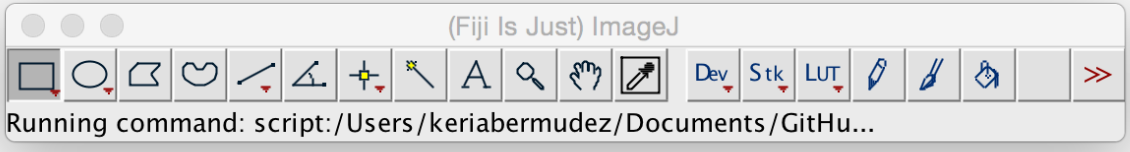
AnalyzeParticlesBatch_2.ijm

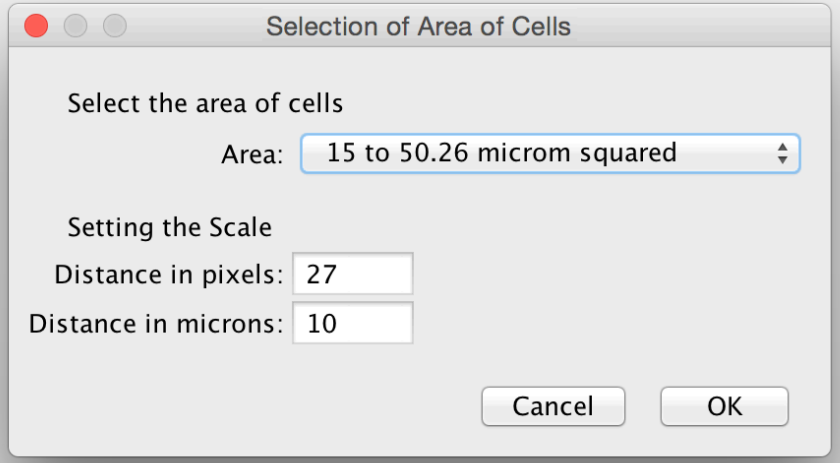
AnalyzeParticlesBatch_2.ijm

```
macro "AnalyzeParticlesBatch_2" {

    //1)Selecting directory of images

    dir = getDirectory("Choose a Directory with images");
    save_path = File.getParent(dir);


    //2)Generate directories to save channels, masks

    File.makeDirectory(save_path+"//"+"Red") ;
    File.makeDirectory(save_path+"//"+"Green");
    File.makeDirectory(save_path+"//"+"ROIs") ;

    //Get list of files in dir
    list = getFileList(dir);

    /*
     * Loop
     */

    for (i=0; i<list.length; i++) {

        if (endsWith(list[i], ".tif")){

        //3)Open RGB image

        open(dir+list[i]);

        //4)Get RGB image name
```

You can do more with macros

- Build a dialogue window





Practice

- Doing the macro yourself
- Do your own macro based on your needs
- Remember to sign a sheet if you want more tutorials on Image analysis