

Example: Atlas-based imaging data analysis
tool for quantitative mouse brain histology
AIDAhisto

Niklas Pallast
Department of Neurology
University Hospital Cologne

2019

This is a step-by-step procedure to count all cells visible in the red channel of a whole brain slice as shown in Figure 1.

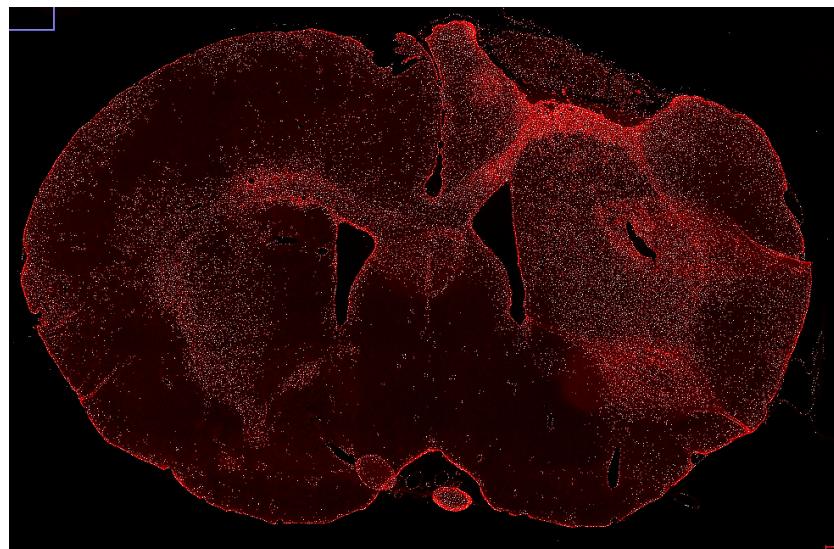


Figure 1: Example microscopy image (red channel): white spots indicate results of cell counting.

1 Download & Install

1. Download the zip-File `AIDAhisto-master` using the following [Link](#).
After opening the web page, the zip-file can be downloaded (Figure 2).

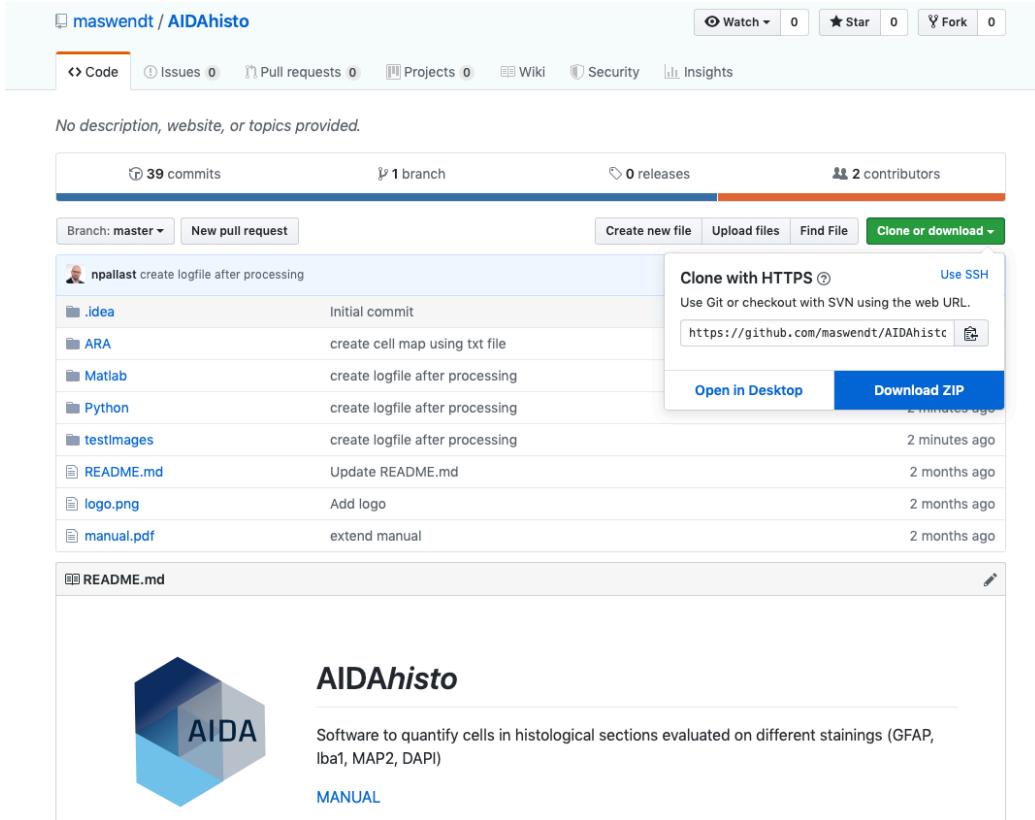


Figure 2: Click on Clone or Download → Download ZIP to download the zip-File.

2. Download & Install Python 3.6 or higher using [Anaconda](#) and enter the following command to install all necessary Python packages
`pip install numpy=1.14.3 argparse=1.4.0 scipy=1.2.1 matplotlib=3.0.3`
3. The code was also implemented in Matlab (tested with version R2018a).
Note: Matlab processing is faster and visualization better compared to Python; otherwise there are no differences.

2 Atlas Transformation

In the provided example files in the folder `.../AIDAhist-master/testImages/wholebrain_atlas.` the transformed atlas slice is already included. In order to perform the atlas transformation, conduct the following steps:

1. Load ARA with highest resolution ($10\mu m$) from the folder
`.../AIDAhisto-master/ARA/annotation_10.nii.gz`
2. Load microscopy image in ImageJ from the folder
`.../AIDAhisto-master/testImages/wholebrain_slice.tif`
3. Select the matching slice in the ARA and make a substack with ImageJ:
Image → Stacks → Tools → Make Substack
4. Choose the corresponding slice number of the ARA and save the substack.
5. Choose "Multi-point" in ImageJ and place 20-40 landmarks in the microscopy image and the corresponding positions in the ARA (Figure 3). Note: that step requires some experience in mouse brain anatomy to match in atlas landmarks with the microscopy - especially if like in this example the mouse brain tissue is strongly deformed due to a local brain lesion.

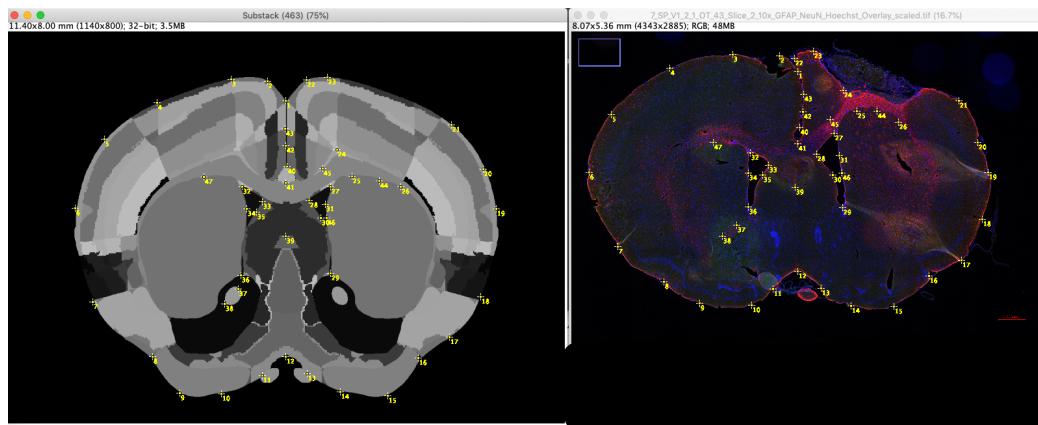


Figure 3: 30-50 landmarks in the microscopy image and the corresponding positions in the ARA

6. Landmark registration with ImageJ: Plugins → Transform → Landmark Correspondences
7. Choose the substack of the ARA as "source image" and the microscopy image as template image in the transform menu
8. Choose: Transformation method: Moving Least Squares (non-linear), Transformation class: Affine, No interpolation (Figure 4).

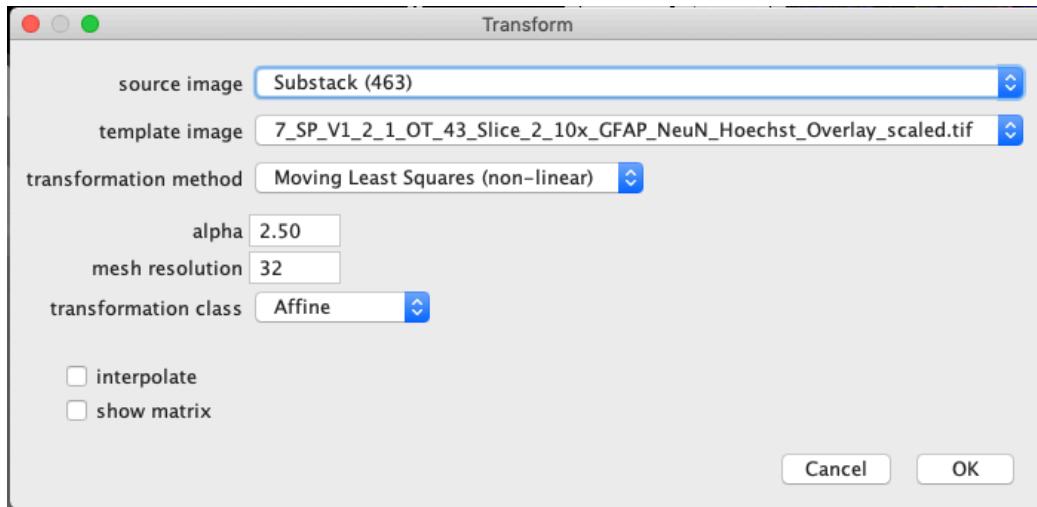


Figure 4: Choose the shown tranformation parameters to register the atlas with the microscopy image

3 Count Cells

1. Count all cells in the given image with Matlab: Here, you have to open Matlab and set `.../AIDAhisto-master/Matlab` as "Current Folder" (Figure 5A). Type the same command in the "Command Window" (see Figure 5B) like shown in figure 6.

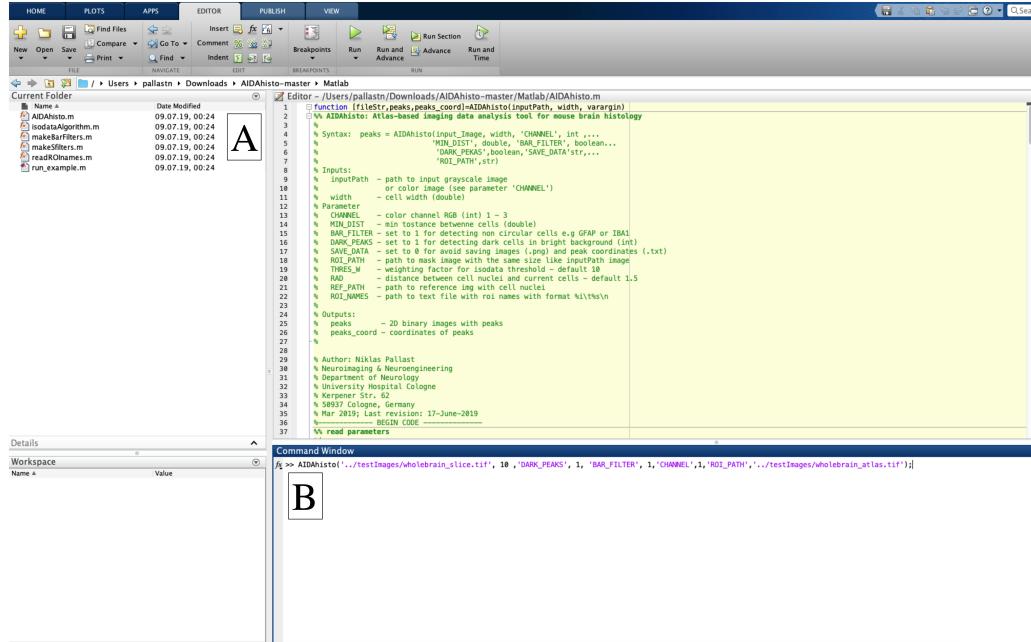


Figure 5: Set the `.../AIDAhisto-master/Matlab` as "Current Folder" (A) and type the command of figure 6 in "Command Window" (B)

```
f1 >> AIDAhisto(''..//testImages/wholebrain_slice.tif'', 10 , 'DARK_PEAKS' , 1, 'BAR_FILTER' , 1,'CHANNEL',1,'ROI_PATH','..//testImages/wholebrain_atlas.tif');
```

Figure 6: Type the shown command in the "Command Window"

2. Count all cells in the given image with Python: Open the Command Prompt if you use a Windows System or Terminal if you use a Macintosh System. Set `AIDAhisto-master/Python` as your current folder using the command `cd` (see figure 7)

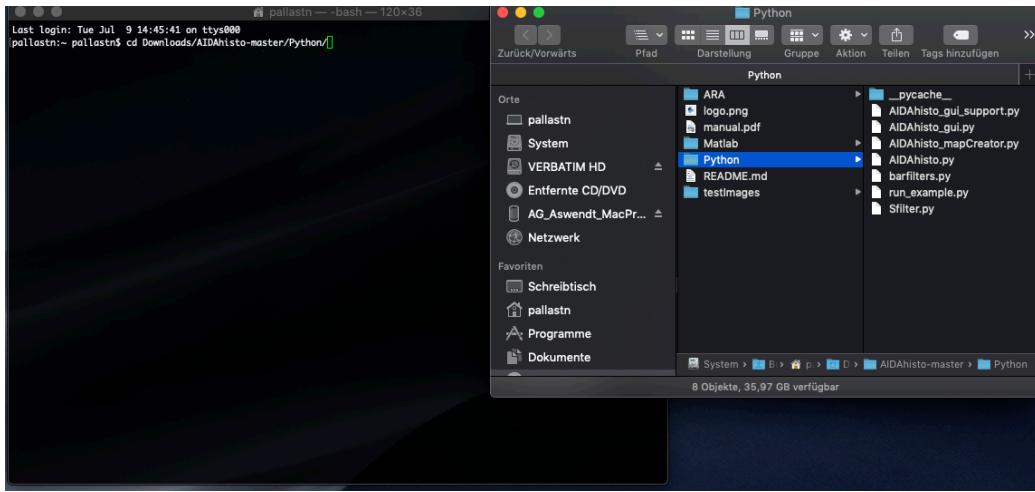


Figure 7: Set the Python folder as your current folder using the command `cd`

3. Type the following command in the command window

```
python AIDAhisto.py ../testImages/wholebrain_slice.tif -f -w  
9 -d -c 0 -a ../testImages/wholebrain_atlas.tif -l ../ARA/acronyms_ARA.txt
```

4. The results are stored in the folder `AIDAhisto-master/testImages` and can be easily overlaid in ImageJ like described [here](#). Note: the counted cells will be represented by a single white pixel, which might be difficult to see in the overlay. Try to use the dilate function in ImageJ on a binarized version of the counting result.