Instructions for Cell Analysis Pipeline

Michael Vaiana

May 31, 2018

Contents

Chapter 1

Getting Started

1.1 Introduction

These notes describe all the steps necessary for a member of the Muldoon lab to go from aligned movie data (obtained from the Goldberg lab) to final analysis. I assume if you are reading this you are familiar with the experimental and research paradigm. Namely that the Goldberg lab is recording calcium imaging movies in mice as they are heated to the point of seizure and that this data is split in to 3 epochs (baseline, early, preictal). If the specifics of the epochs change in the future simply adjust accordingly.

The basic steps or processing and analysis are as follows, and the remainder of the notes just adds details to these steps.

1. Check if the movies are aligned (z-plane) by looking at averages over many frames for each epoch. (imagej)
2. Update a spreadsheet indicating which movies are aligned and usable
3. Detect ROIs on the images which are well aligned (imagej)
4. Extract cell traces from the movie (MATLAB)
5. Determine time of seizure and thus frame times for each epoch (MATLAB)
6. Update the spreadsheet with the frame times
7. Save final form of DF/F traces based on frame times (Python)
8. Extract spiking and features and save this data (Python)
9. Measure population statistics (Python)
10. Use Mapper GUI (Python)
11. Multilayer Community Detection (MATLAB)

The reason for the mix of Python and MATLAB is because I found premade tools in each which do a specific job. In MATLAB there is free code (twophotonanalysis) which will take a movie file name and an ImageJ roi file name and extract the trace for each cell in the ROI file. Python is necessary to use the OASIS package for spike inference. MATLAB is necessary to use GENLOUVAIN community detection code. A nice project would be to streamline this whole process into either Python or MATLAB by transferring code from one to the other.

1.2 Installation

Below I will give detailed instructions for how to get all the code you will need to run the pipeline. These instructions are not the absolute only way to install this so if you know what you are doing feel free to do it another way.

Python

First install anaconda `https://www.anaconda.com/download/`. Then follow the code below from within the directory you want to install these files (for example ∼/repos/).

Clone the python repos we will use

```
git clone https://github.com/j-friedrich/OASIS.git
git clone https://github.com/vaiana/caspan.git
git clone https://github.com/vaiana/dmaps.git
```

Create Analysis Environment

```
conda create --name calpipe python=3 numpy scipy pandas matplotlib nb_conda cython pip
source activate calpipe
cd OASIS
pip install -e .
cd ../caspan
pip install -e .
cd ../dmaps
pip install -e .
cd ..
```

You should now have a working environment for the analysis pipeline. At the beginning of an analysis session you should type `source activate calpipe` to activate this environment before running any of the jupyter notebooks.

Create Mapper Enviornment

Deactive the calpipe environment by `source deactivate`. Now follow the instructions below to make an environment for Mapper.

```
conda create --name mapper python=2 numpy scipy matplotlib wxpython=3 graphviz pyopengl pip
source activate mapper
pip install fastcluster
pip install mapper
pip install cmappertools
```

Test Mapper Installation

MAC

```
cd /anaconda3/envs/calpipe/bin
pythonw MapperGUI.py
```

Linux

```
cd ∼/anaconda3/envs/calpipe/bin
python MapperGUI.py
```

Windows

```
?
```

MATLAB

You will need to install Genlouvain and MultiLayerNetworks. When doing MATLAB analysis make sure these are on your path.

Clone the repos

```
git clone https://github.com/GenLouvain/GenLouvain.git
git clone https://gitlab.com/vaiana/multilayer-networks.git
```

Analysis and Processing

All of the data (except raw movie files), code (except the installations above) and notes for the calcium imaging analysis are kept in a repository called capipeline.

Clone the analysis and processing directory

```
git clone https://gitlab.com/vaiana/capipeline.git
```

You should now have a directory (/repos/capipeline). Navigate to this directory and look around to familiarize yourself with its structure. There are 2 directory for code `matlab` and `notebooks` the later for Jupyter notebooks where Python analysis takes place.

Chapter 2

Data Processing

2.1 Summary File

Everything is controlled by `/repos/capipeline/summary.csv`. There is also `/repos/capipeline/summary.ods` which can be opened in the free program LibreOffice or can probably be opened in Microsoft Excel. I typically edit the .ods version because its visually nice (color coded) and then save a copy to `summary.csv`.

The most important column is the column `use` which indicates if that mouse should be used or not. The other important columns are `baseline_start, baseline_end,` etc. which determine the frame numbers of the start and end of each epoch. The frame numbers are determined as if the baseline, early, preictal and ictal periods have all be concatenated into one long movie. These frames are determined later in the pipeline. All the code works by reading the `summary.csv` file and using the mice indicated (`use =1`) and then using the mouse id and session to read and write files.

2.2 Naming Conventions

The data we receive is from the Goldberg lab. The data naming and structure has been somewhat inconsistent for a variety of reasons so we standardize everything on our end. In general the names of files are of the form

`m{mouse_id}_{session}_{additional_info}.{ext}`

For example, the ROI file for mouse 18 in the first recording sesion is `m18_01_all.zip` where the all indicates that the ROIs are good for all epochs. The file names must match the data in the spreadsheet. For example, to read the movie file `m963_03_preictal.tif` the mouse id column must read 963, the session must be 3 (or 03 - the 0 gets added automatically in the code if its missing).

IMPORTANT! These naming conventions must be followed since the pipeline automatically reads data based on file names

Movie Names

The movies are stored on a data drive `/data/ca_seizure/movies` on Mike's school computer, if you move/copy this data adjust the instructions accordingly. To each mouse there are 3 or 4 movies corresponding to the epochs: baseline, early, preictal, ictal(may not be present). The movies MUST follow this naming convention:

`m{mouse_id}_{session}_{epoch}.tif`

For example, if the preictal epoch for mouse 963 in recording session 2 (the second time this mouse has been imaged) then the file name must be `m963_02_preictal.tif`

ROI Names

Rois found in `/data/ca_seizure/roi`

The ROI names MUST follow: `m{mouse_id}_{session}_all.zip`

2.3 Raw Movies

The raw movies should be opened in ImageJ and all epochs should be averaged and concatenated into a single stack to determine alignment. If aligned ROIs can be detected on this concatenated stack via the ROI Manager in ImageJ. Save the concatenated stack in `/data/ca_seizure/images/` with a descriptive name and save all the ROIs in `/data/ca_seizure/roi/` with the naming convention described above.

2.4 Traces and Spikes

MATLAB After ROIs are detected raw traces and DF/F data is extracted in MATLAB. Use

`capipeline/matlab/preprocess/make_trace.m` and

`capipeline/matlab/preprocess/make_dff.m`

to compute and save the raw and DF/ F traces. DF/F is computed with `compute_dff.m` which gives different options for computing baseline but in general the baseline is the mean of the bottom $X\%$ where X is given in the `make_dff.m` file. Raw traces may take some time to be calculated. The file `make_trace.m` has an `overwrite` variable which is set to 0. This means that the traces will NOT be recomputed if they are already there, but if new data is added the new data will be computed, this saves some time if you are adding in extra mice since the old data won't be recomputed. If you need to recompute old traces (maybe ROI changed?) then set the overwrite variable to 1.

Once the traces and spikes are created use `capipeline/matlab/preprocess/plot_mean_activity.m` to view the average DF/F over all epochs to determine the seizure start frame. Once this is done edit the `summary.csv(.ods)` file to indicte the frame times for each period. The lastest movies seem to have the seizure very close to the beginning of the 'ictal' movie so these frame numbers might correspond exactly to the first 24000 frames (8000 per epoch). See the `summary.csv` file for examples.

PYTHON Use `capipeline/notebooks/save_final_traces.ipynb` to slice the DF/F traces into their final form (according to the indices in `summary.csv`) and save these in `capipeline/traces/final`.

```
(assumes source activate calpipe)
cd  /repos/capipeline/notebooks
jupyter notebook
```

Then open `save_final_traces` and 'run all'.

Next we use oasis to deconvole and save spikes and features. Within the browser, go to the jupyter home page (directory) and open `preprocess` run all to save spikes and features.

2.5 Summary

We used MATLAB and Python to save spikes and features for further analysis. The analysis we will do is population statistics on the features (Python), mapper analysis (Python), and multilayer community detection (Matlab).

Chapter 3

Analysis

Both Python and Matlab are used for analysis. The Python code is contained in `capipeline/notebooks` and the Matlab analysis code is contained in `capipeline/matlab/analysis`. The first step should probably be using `capipeline/notebooks/population_analysis` to measure the population features. This basic global analysis will suggest some trends in the data although it is good to look at the mice individually (also in the same notebook) to determine if the results are actual trends or being driven by outliers. Essentially with our data so far it seems that the trends are not consistent.

IMPORTANT! Always activate the calpipe environment before doing any Python analysis in the Jupyter Notebooks

```
source activate calpipe
```

IMPORTANT! In the current version the wild type only have baseline and early periods and the preictal period is essentially bogus (the cells have moved) so only use baseline-early for any interpretation

Population Analysis

Run `capipeline/notebooks/population_analysis`. This will compare population features of the 3 epochs. It will also compare wild type to treatment groups. T-test p-values and bar-charts are generated.

Mapper Analysis

The mapper analysis requires the mapper GUI. First activate the mapper enviorment

```
   source deactivate # only neccessary if another environment is active
source activate mapper
```

Open Mapper GUI

MAC

```
cd /anaconda3/envs/calpipe/bin
pythonw MapperGUI.py
```

Linux

```
cd ~/anaconda3/envs/calpipe/bin
python MapperGUI.py
```

Use Mapper GUI

1. Click 'Load Data' tab
2. Select the data you want to use, for the thesis I used `basepre_max` which is the differences in baseline and preictal with each feature (column) normalized by the maximum.
3. Select Parameters
    a) Metric: Euclidean

b) Filter Function: Distance Matrix Eigenvector, NOT mean-centered, 0th eigenvector

c) Uniform 1-D cover: Play with number of intervals (coarsness, overlap 50% should be fine)

d) Clustering: Single, First Gap, 0.1, but Play around with these also, especially gap-size

e) Node Coloring: `point_color = data[:,0] # change 0 to 1,2,3, etc. it is the column and the nodes will be colored by the values in that column. For example, if burst-amplitude is the first column then data[:,0] will color the nodes by the average burst amplitude of the data points contained in the node.`

Now run mapper and save the nodes in .txt files to keep track of which cells belong to which nodes in the graph. You can click nodes (hold shift to get more than one) and then ctrl-s to save the nodes.

Once the nodes are saved use Python `capipeline/notebooks/mapper_analysis` to pull them in and look at the traces as well as compute some overall statistics of the groups (for example mean).

Community Detection

The community detection is done in Matlab with `capipeline/matlab/analysis/mln_analysis.m`. This code should run and display the community detection results (sorted by layer 1) as well as the raster plots (sorted by layer 1). Even though the axes are labeled upside down (relative to each other) the cells are consistent (the first cell is always at the top). Run it and you will see what I mean. The plots are labeled by which mouse it came from and if its wild that is also in the plot title. You will need to have the following folder on your path.

```
capipeline/matlab
/repos/Genlouvain
/repos/multilayer-networks
```

As a side note, the plugin folder: twophotonanalysis, needs to be named this because the plugin references the directory name.