

Block course

Day 1: Github, python and [Ca] data binarization

Catalin Mitelut, PhD
Donatolab

Overview

1. Using **Anaconda** as an environment manager (quick intro)
2. Using **Github** as algorithm repository (quick intro)
3. **Python** (quick intro)
4. **Binarizing** Suite2p data using python

Quick polls

How many of you have coded before?

What about python or matlab?

How many heard of conda, or anaconda or virtual environments?

What about git and github?

(Ana)Conda

(Ana)conda Intro

Conda is a “package manager”

- Helps us install and manage functions, package etc in a coding environment
- What is an environment? (hint: what is a native environment?)

So we can make our own “environment” that remembers:

- version of python we install
- versions of other packages we install
- ... and many other settings

For today we already have our own environment for the course:

block_course

To load it

- We use windows anaconda command line and type:
conda activate block_course

Creating an environment with commands

Tip

By default, environments are installed into the `envs` directory in your conda directory. See [Specifying a location for an environment](#) or run `conda create --help` for information on specifying a different path.

Use the terminal or an Anaconda Prompt for the following steps:

1. To create an environment:

```
conda create --name myenv
```

Note

Replace `myenv` with the environment name.

2. When conda asks you to proceed, type `y`:

```
proceed ([y]/n)?
```

This creates the myenv environment in `/envs/`. No packages will be installed in this environment.

3. To create an environment with a specific version of Python:

```
conda create -n myenv python=3.9
```

4. To create an environment with a specific package:

```
conda create -n myenv scipy
```

OR:

```
conda create -n myenv python
conda install -n myenv scipy
```

5. To create an environment with a specific version of a package:

```
conda create -n myenv scipy=0.17.3
```

OR:

```
conda create -n myenv python
conda install -n myenv scipy=0.17.3
```

Github

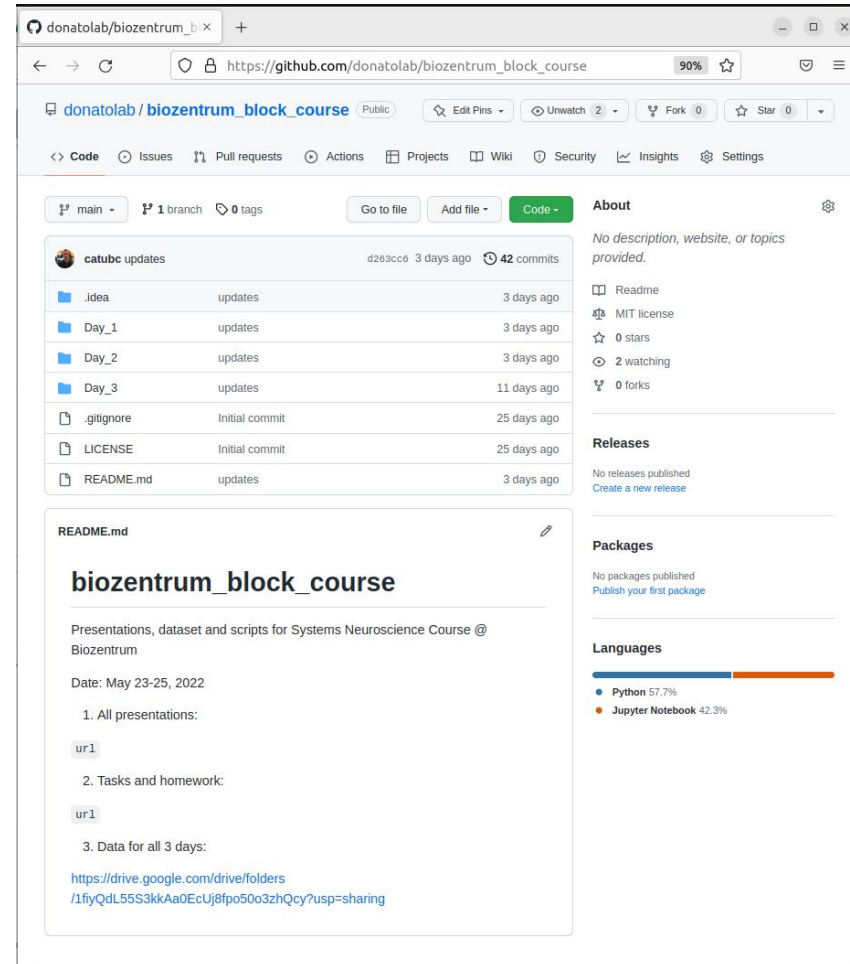
Github Intro

Github is also several things

- Today we focus on repository storage and retrieving
- But what is a repository ?

Git vs. github

- Git is a “distributed version control system”
 - Allows teams of developers to write code together each contributing small parts
 - Allows retrieval and management of different version of the code (e.g. retrieving old code when made a mistake)
- Git can be run locally on single computer (or group)
- Github is essentially a managed git service hosted publicly at github.com that can be accessed by anyone
 - Can have both public and private repos



Python

Interface for coding in python

- We use Jupyter-lab or Jupyter notebook which are dynamic/interactive
 - Question: What does “interactive” mean?

Variables

- String
 - `fname = r"C:\Users\block_course.npy"`
- Scalar
 - `x = 5`
- Vectors
 - `x = [0,1,2,3,4]`
- Matrices
 - `x = [[0,1,2],
[3,4,5],
[6,7,8]]`
- Objects
 - `c = Calcium()`

Many intrinsic packages and functions (e.g. “numerical python”)

- `import numpy`
 - `x = numpy.arange(5)`
 - `print (x)`
`[0,1,2,3,4]`

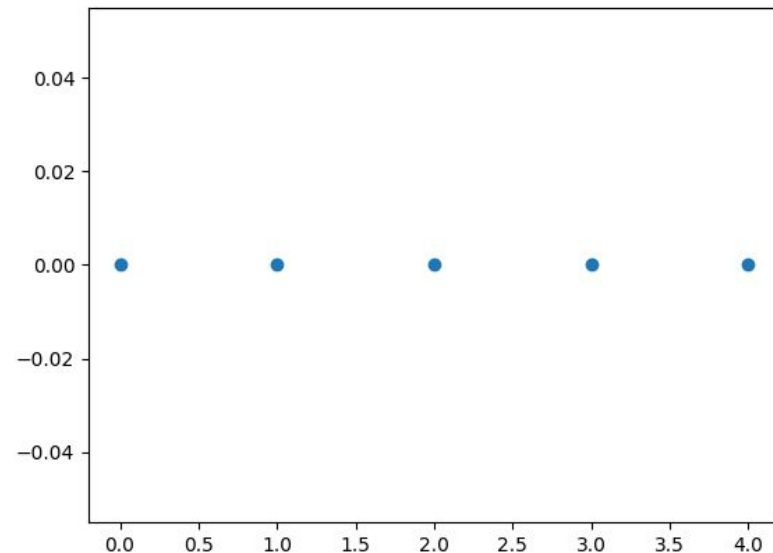
Plotting

- We use matplotlib package to plot a scatter plot

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 # generate a simple array
5 x = np.arange(5)
6 y = np.zeros(5)
7
8 #
9 print ("x: ", x)
10 print ("y: ", y)
11
12 # make a figure
13 plt.figure()
14
15 # scatter plot x vs. y
16 plt.scatter(x,y)
17
18 # show the image
19 plt.show()
```

x: [0 1 2 3 4]

y: [0. 0. 0. 0. 0.]



Using Python to make our own packages (or classes)

- E.g. Calcium class developed @Donatolab
 - We can load the Suite2p output

```
# initialize calcium object and load suite2p data
c = calcium.Calcium()
c.verbose = True # output
c.recompute_binarization = True # recompute
c.data_dir = data_dir
c.load_suite2p()
```

- And we can run some algorithms on the data

```
#####
##### RUN BINARIZATION STEP #####
#####
#
c.binarize_fluorescence()
```

Binarization

Using the Calcium() python class to convert Suite2p output to binarized rasters

1. Provide link to suite2p file location

2. Set parameters

- Threshold: usually $2.5 \times \text{std}$
- [Ca] low pass filter: 0.5hz
- Detrend model degree: 1

3. Output saved as .mat and .npz files

```
#####
##### LOAD SUITE2P AND BINARIZE #####
#####

# input directory where Suite2p outputs matlab Fall.mat and all .npy files
# data_dir = '/media/cat/4TB/donato/steffen/DON-004366/20210228/'
# data_dir = '/media/cat/4TB/donato/steffen/DON-004366/20210301/'
data_dir = '/media/cat/4TB/donato/DON-003343/DON-003343_20210213/suite2p/plane0/'
# data_dir = '/media/cat/4TB/donato/nathalie/plane0'
# data_dir = '/media/cat/4TB/donato/renan/renan'
# data_dir = '/media/cat/4TB/donato/steffen/DON-004366/20210228' # can also add suite2p/plane0/

# initialize calcium object and load suite2p data
c = calcium.Calcium()
c.verbose = True # outputs additional information during processing
c.recompute_binarization = True # recomputes binarization and other processing steps; False: loads from previous saved locations
c.data_dir = data_dir
c.load_suite2p()

# set flags to save matlab and python data
c.save_python = True # save output as .npz file
c.save_matlab = True # save output as .mat file

#####
#### PARAMETERS FOR RUNNING BINARIZATION ####
#####
c.min_width_event_onphase = c.sample_rate//2 # set minimum width of an onphase event; default: 0.5 seconds
c.min_width_event_upphase = c.sample_rate//4 # set minimum width of upphase event; default: 0.25 seconds

##### PARAMETERS TO TWEAK #####
# 1. Cutoff for calling something a spike:
# This is stored in: std_Fluorescence_onphase/upphase: defaults: 1.5
# higher -> less events; lower -> more events
# start at default and increase if data is very noisy and getting too many noise-events
c.min_thresh_std_onphase = 2.5 # set the minimum threshold for onphase detection; default: 2.5
c.min_thresh_std_upphase = 2.5 # set the minimum threshold for upphase detection; default: 2.5

# 2. Filter of [Ca] data which smooths the data significantly more and decreases number of binarized events within a multi-second [Ca]
# This is stored in high_cutoff: default 0.5 to 1.0
# The lower we set it the smoother our [Ca] traces and less "choppy" the binarized traces (but we lose some temporal precision)
c.high_cutoff = 0.5

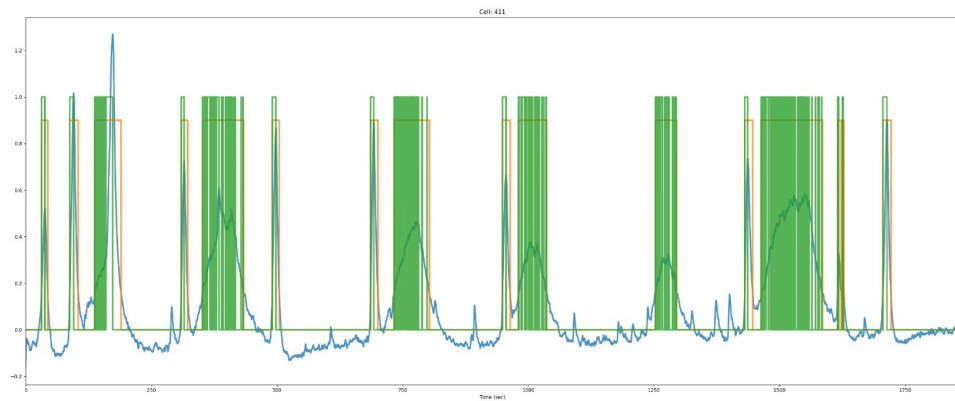
# 3. Removing bleaching and drift artifacts using polynomial fits
# This is stored in detrend_model_order
c.detrend_model_order = 1 # 1-5 polynomial fit

#####
##### RUN BINARIZATION STEP #####
#####
#
c.binarize_fluorescence()
```

Visualizing the binarization code results

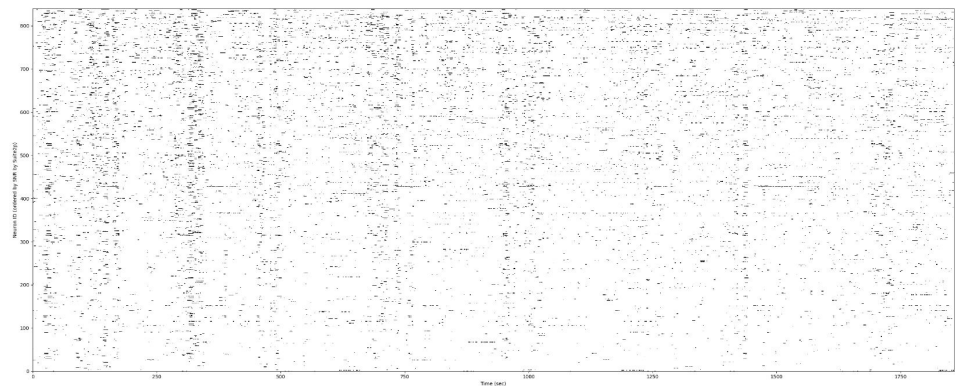
1. Visualize single cell binarization

```
#####  
##### LOAD BINARIZED DATA AND VISUALIZE CELL ACTIVITY #####  
#####  
#fname = '/media/cat/4TB/donato/DON-003343/DON-003343_20210213/suite2p/plane0/binarized_traces.npz'  
#fname = '/media/cat/4TB/donato/steffen/DON-004366/20210228/binarized_traces.npz'  
#fname = '/media/cat/4TB/donato/nathalie/plane0/binarized_traces.npz'  
#fname = '/media/cat/4TB/donato/renan/renan/binarized_traces.npz'  
#  
fname = '/media/cat/4TB/donato/steffen/DON-004366/20210228/binarized_traces.npz'  
  
c = calcium.Calcium()  
c.fname = fname  
c.load_binarization()  
  
# pick a random cell to visualize  
cell_id = 307  
scale = 100  
  
#  
c.plot_cell_binarization(cell_id, scale)
```

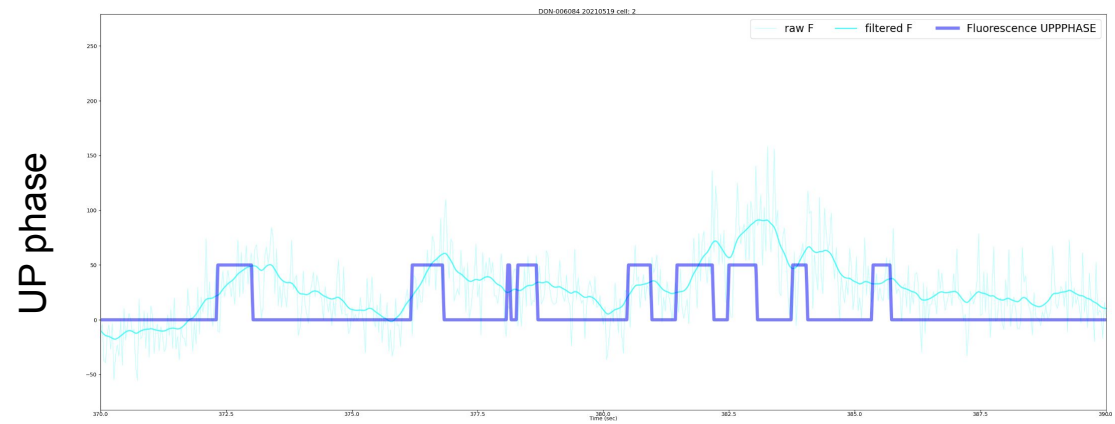
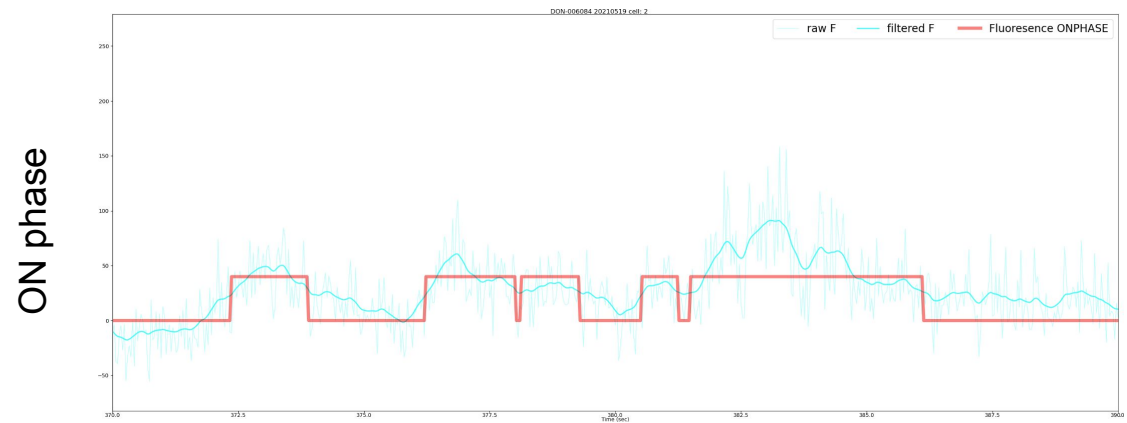


2. Look at entire raster

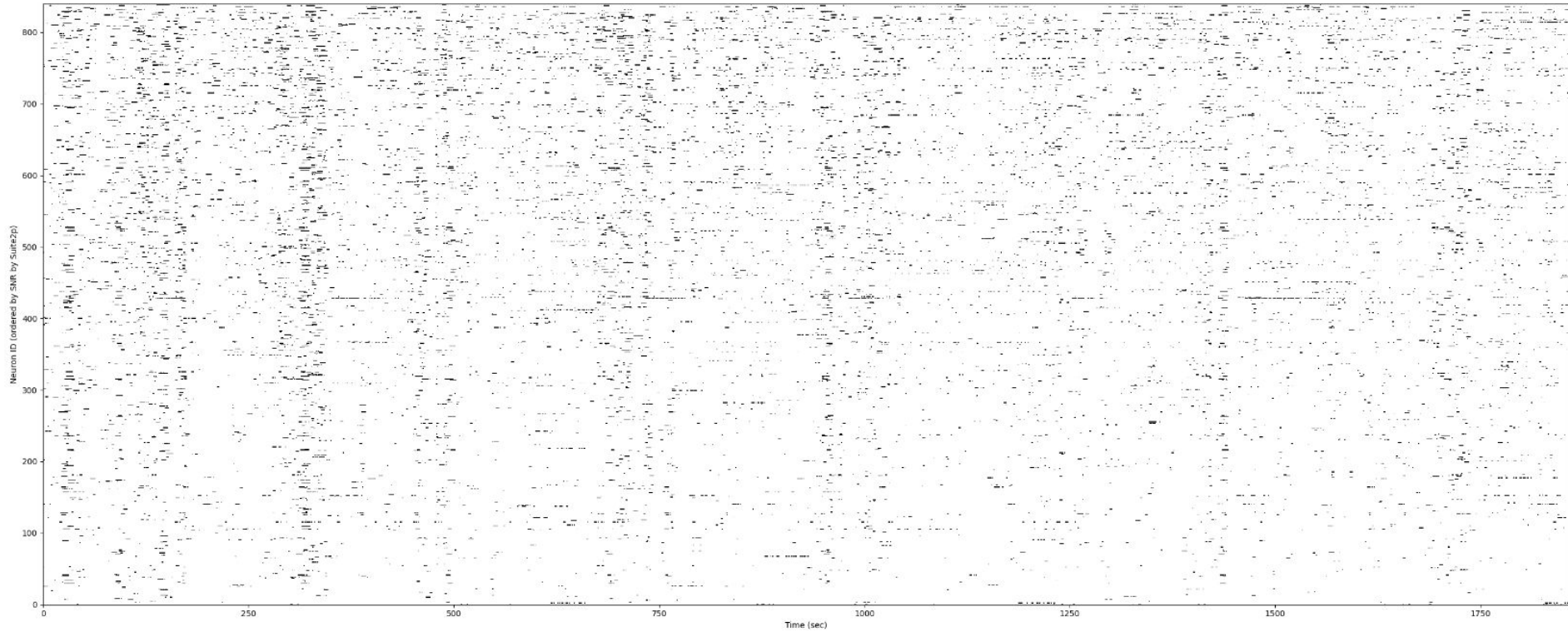
```
#####  
##### SHOW RASTERS #####  
#####  
fname = '/media/cat/4TB/donato/DON-003343/DON-003343_20210213/suite2p/plane0/binarized_traces.npz'  
data_dir = '/media/cat/4TB/donato/DON-003343/DON-003343_20210213/suite2p/plane0/'  
  
c = calcium.Calcium()  
c.data_dir = data_dir  
c.fname = fname  
c.recompute_binarization = False  
c.load_binarization()  
  
c.show_rasters()
```



Binarization: converting [ca] activity into spiking/boolean data

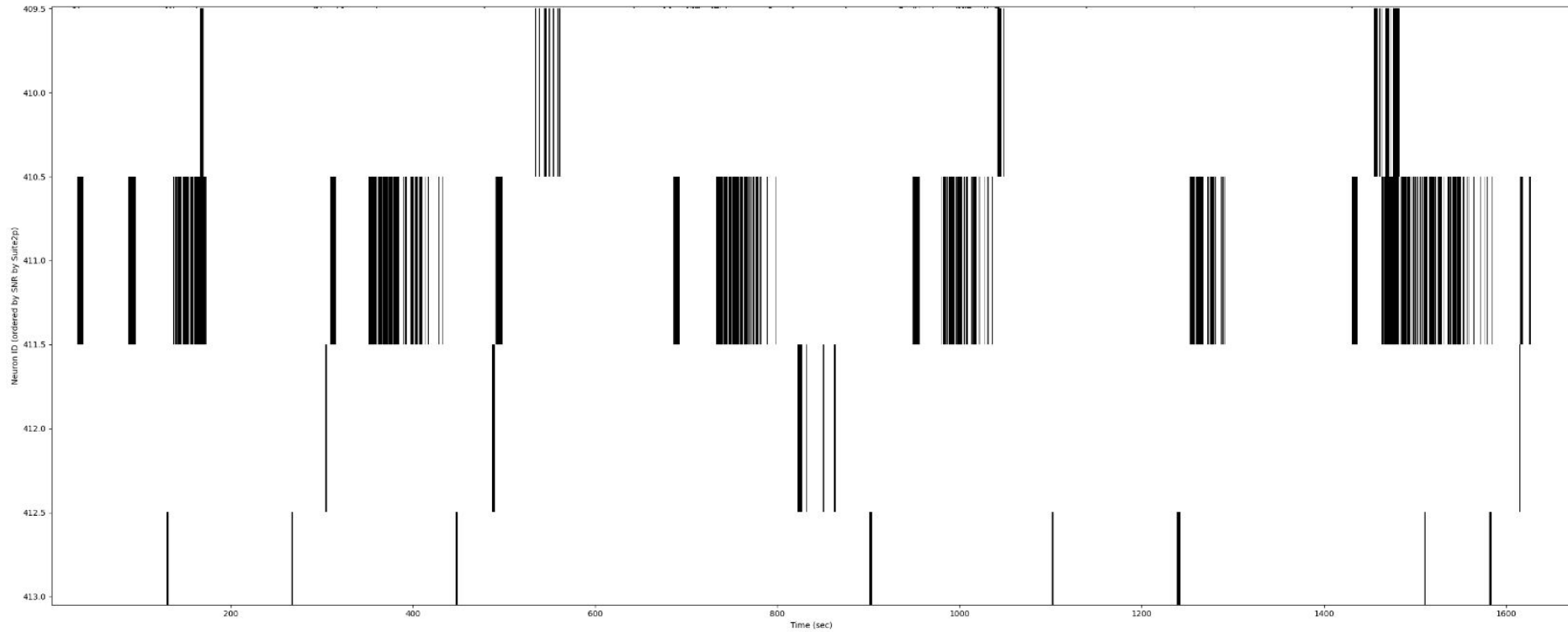


Example of intrinsic neural activity (mouse 3343)



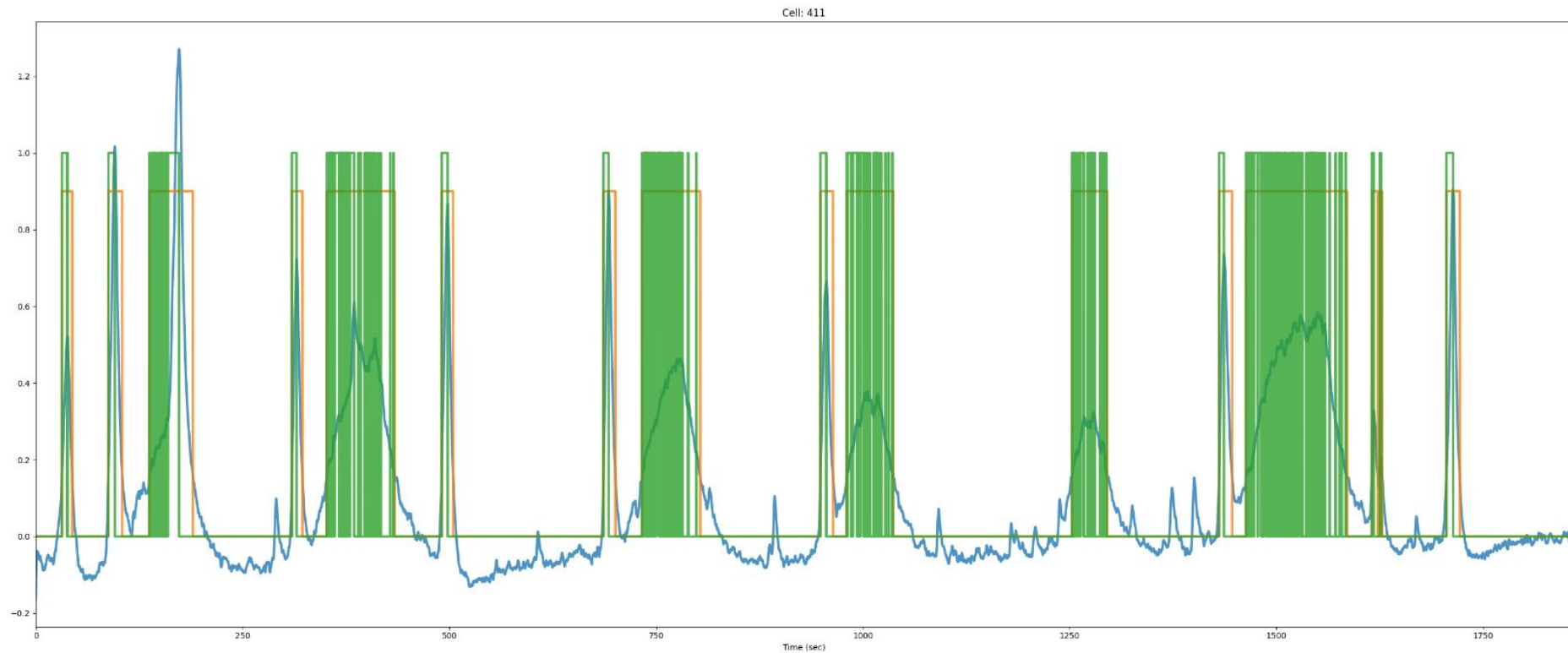
Q: Are there any weird / bad cells??

Zooming in on raster of potentially bad cell (using python)

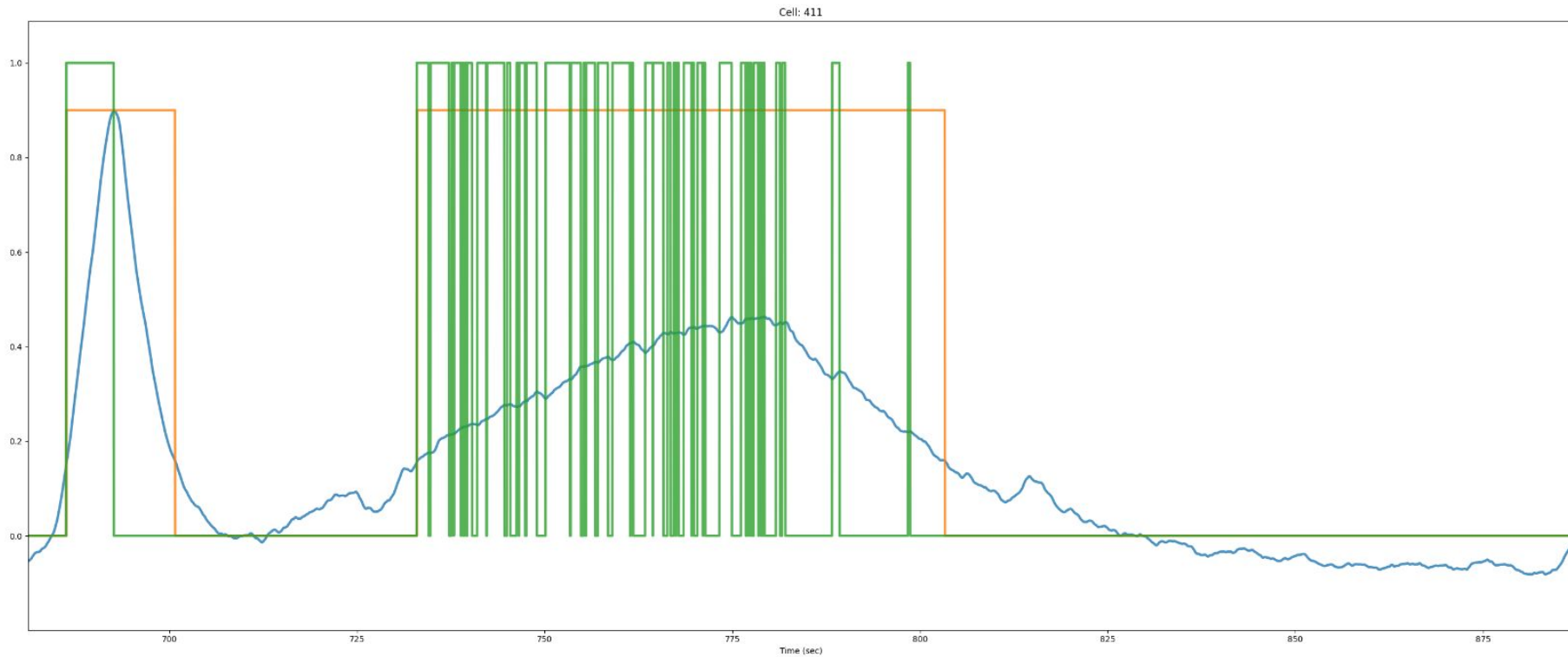


Q: what is the neuron id of the weird / bad cell?

We can then visualize the bad/weird cell

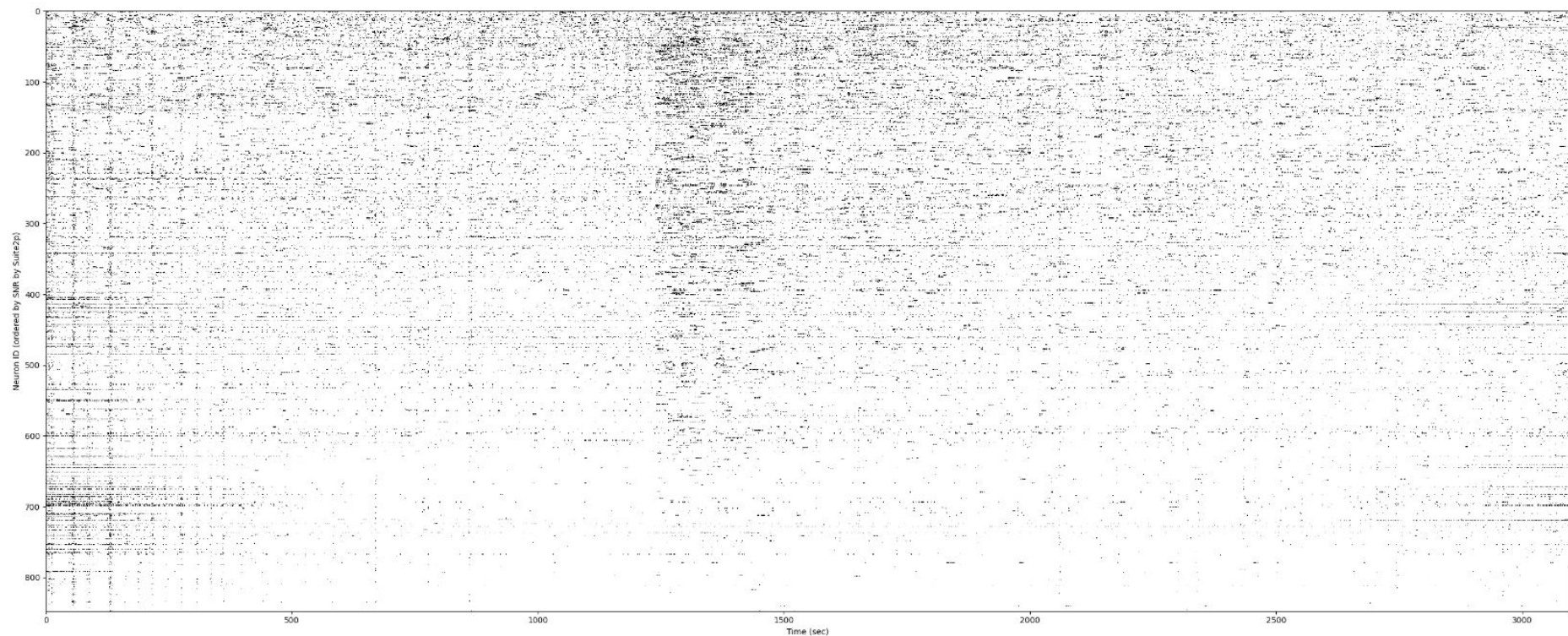


We can also zoom in even more to look at the UP-states found by our algorithm



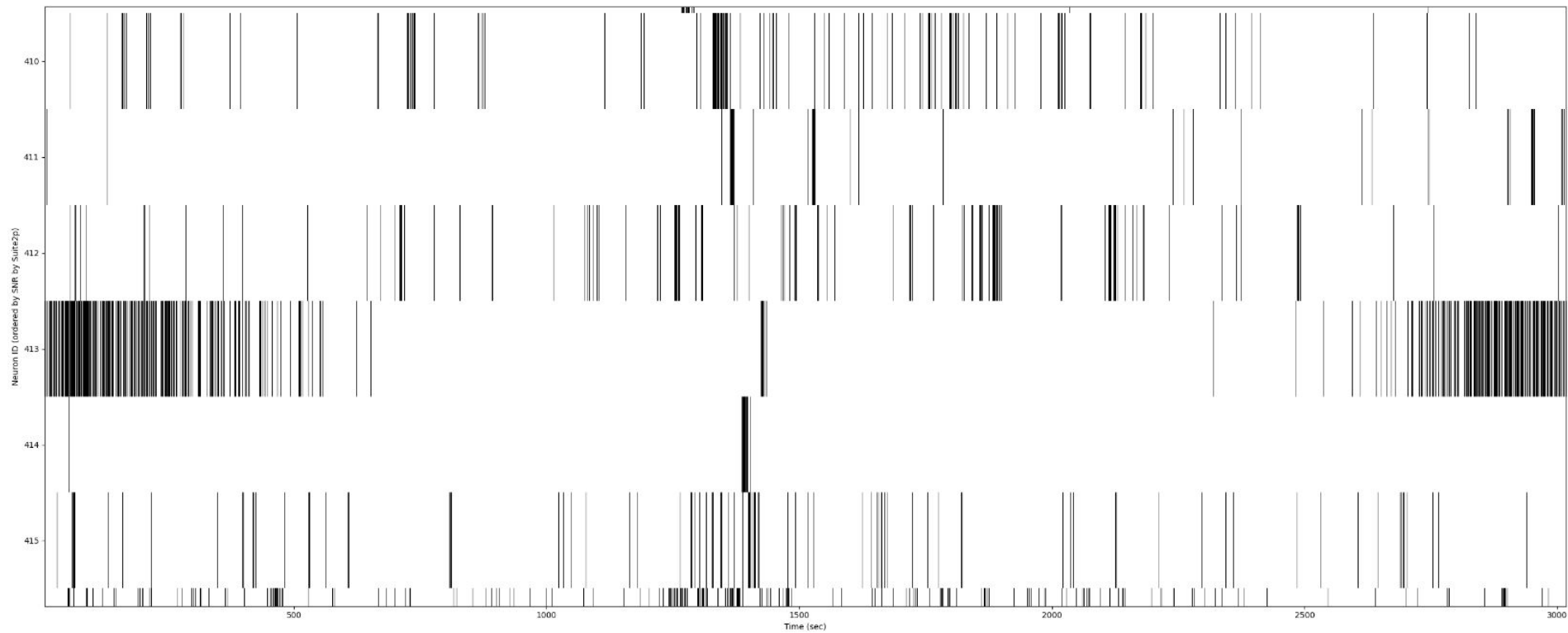
So cell... looks a bit bad

Spontaneous foraging neural data (1P)

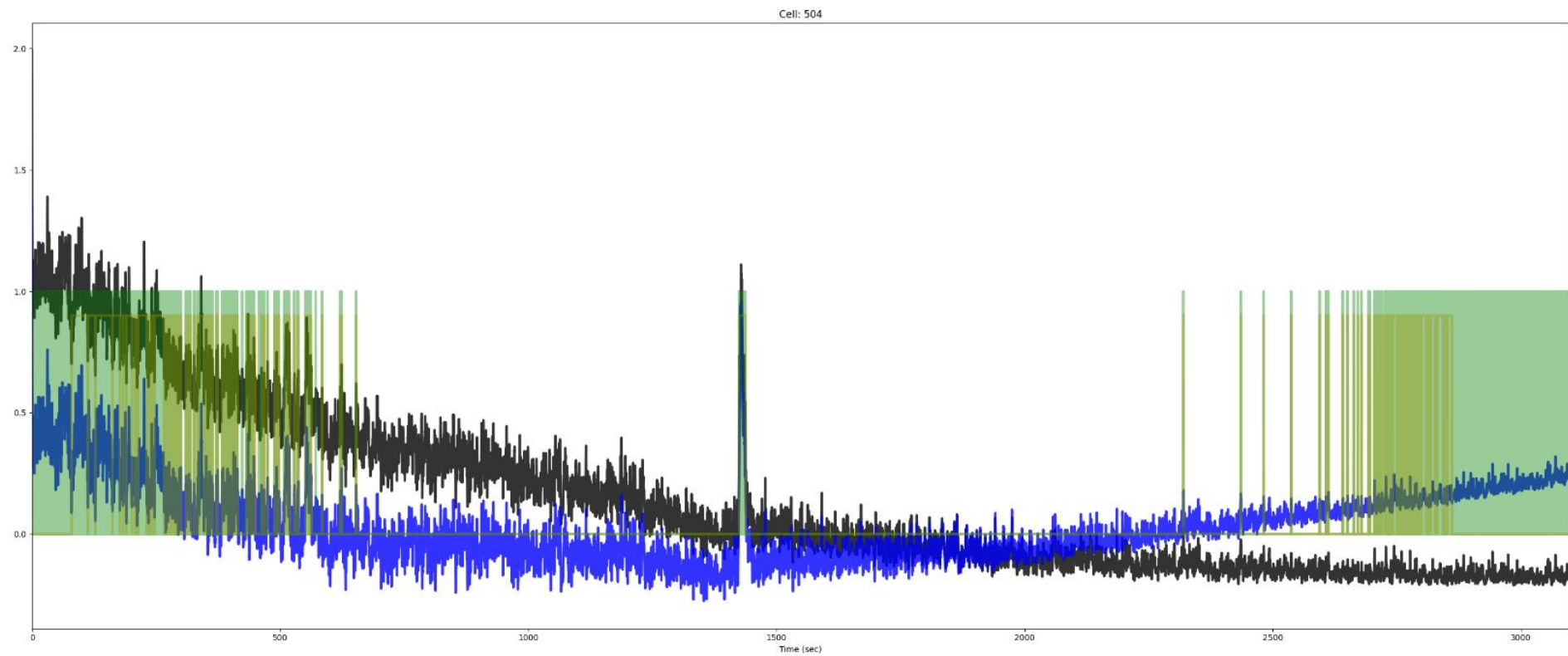


Lots of weird cells!!

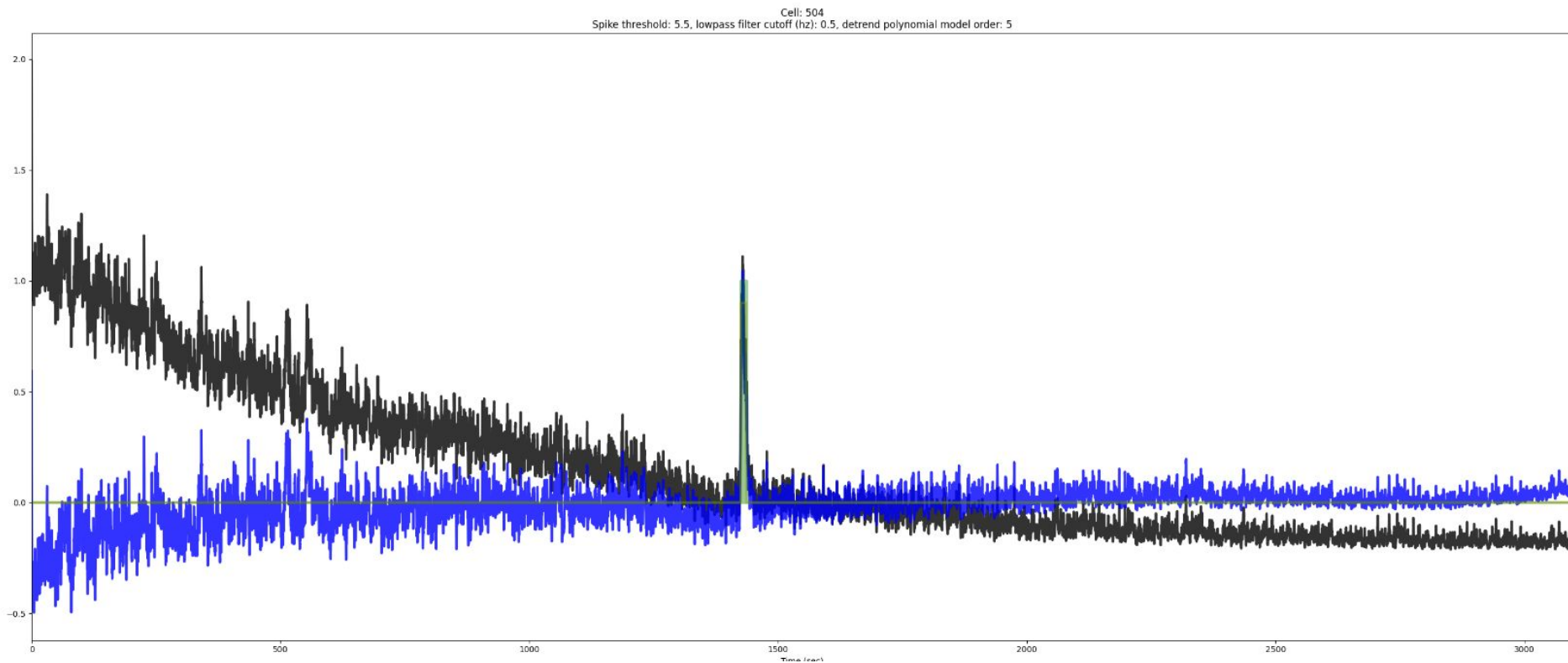
Steffen treadmill data



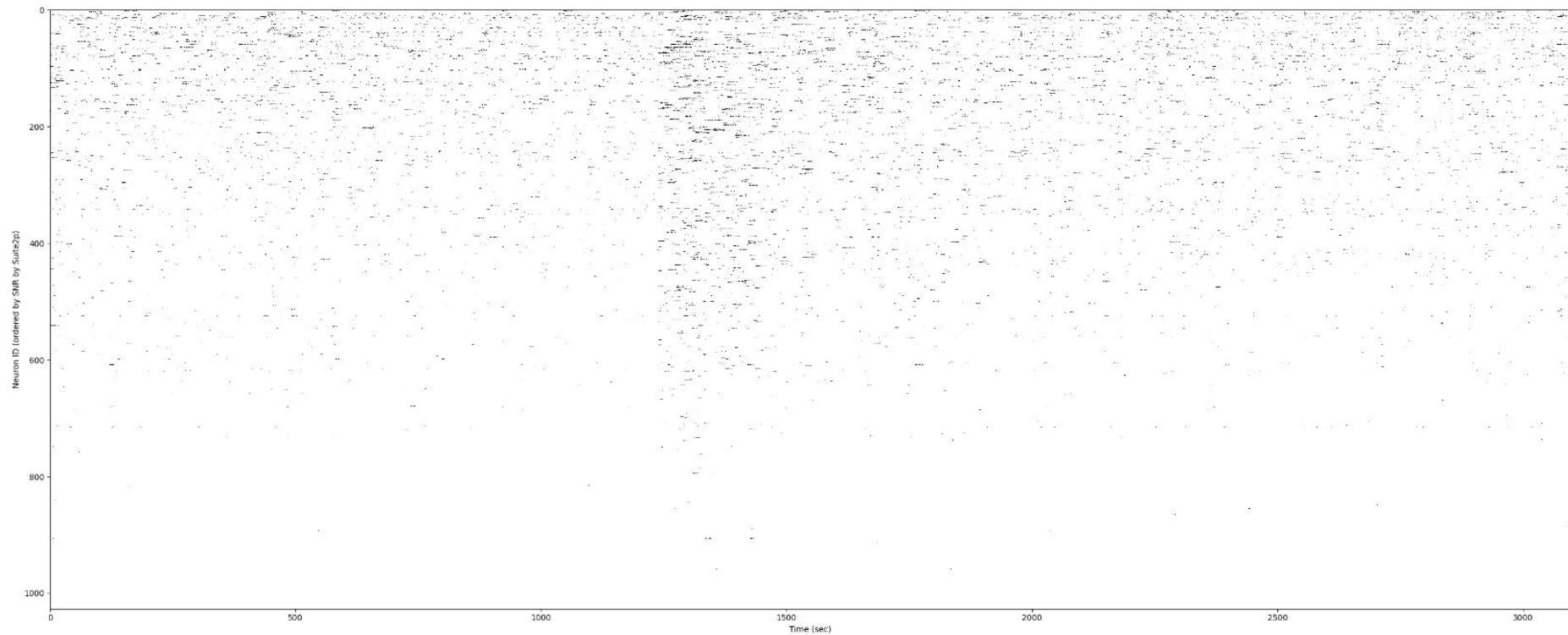
Cell has bleaching (?!)



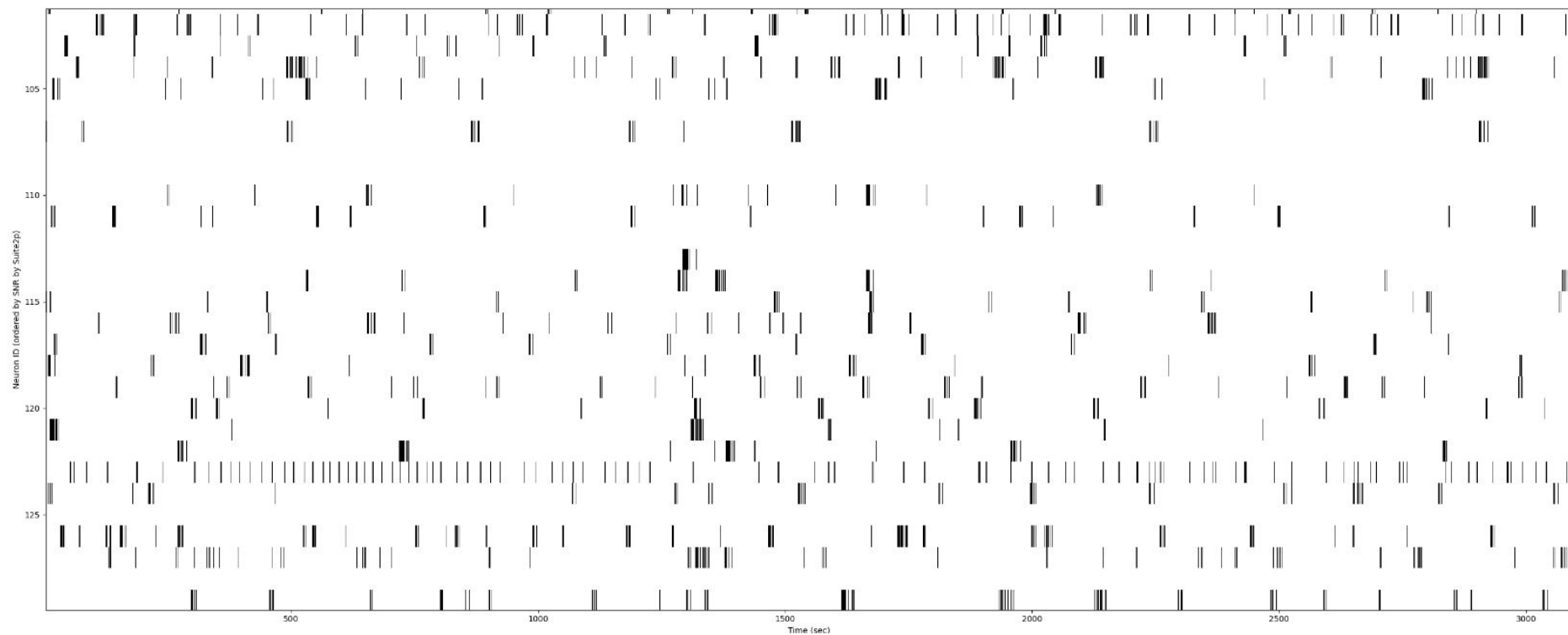
Improving cell dynamics using - higher order polynomial fit



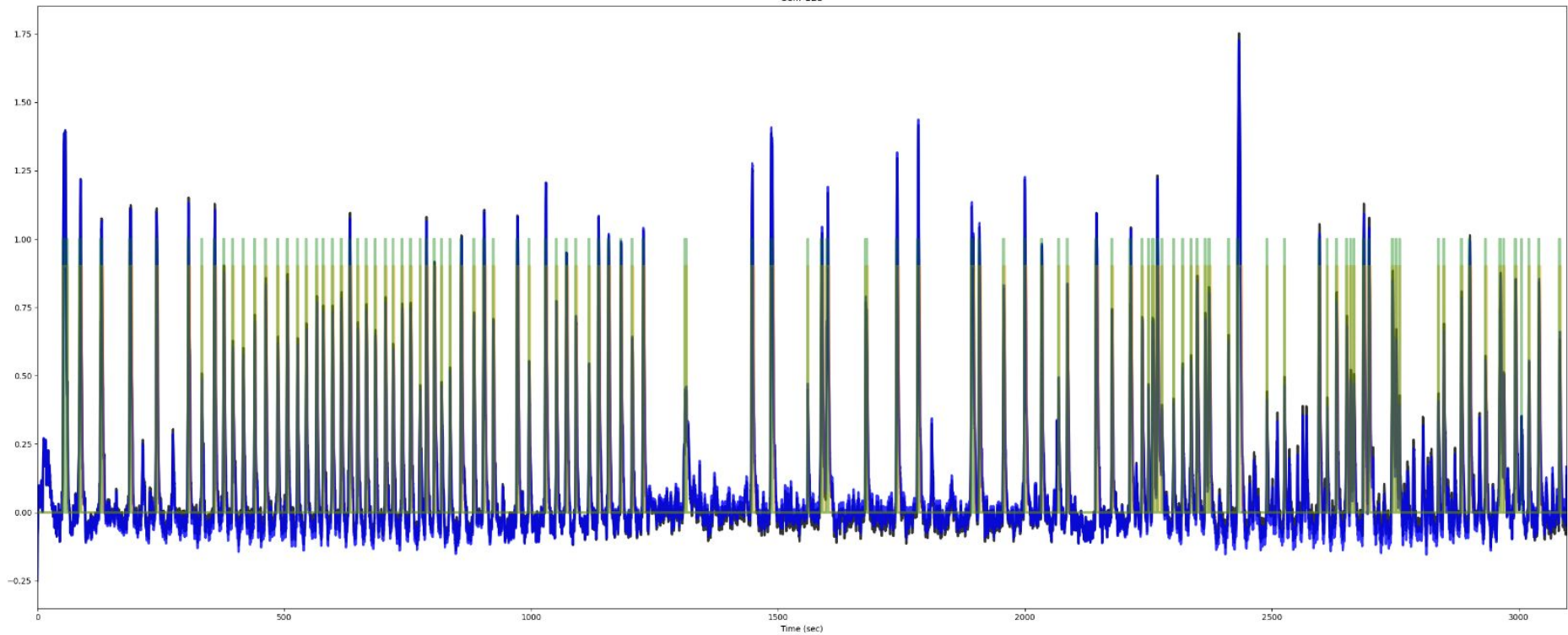
Spontaneous foraging neural data (1P) - looks much better



Lastly - treadmill task (mouse does laps over a cued treadmill)

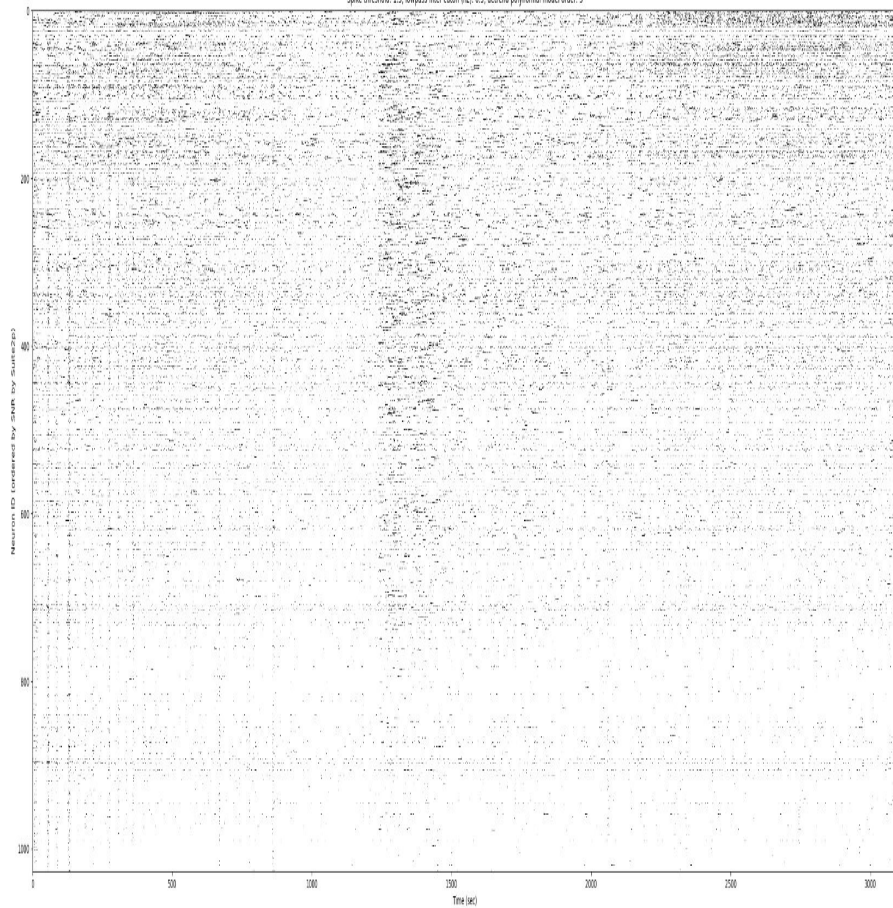


Cell: 123



Thresh: 1.5

Spike threshold: 1.5, lowpass filter cutoff (Hz): 0.5, detrend polynomial model order: 5



Thresh: 5.5

