analyze_batch

January 13, 2020

[1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import pysta
import stc
%load_ext autoreload
%autoreload 2
```

1 run for all cells (OFF LINE)

run

`python3 stc_batch.py [DATASET]`

datasets * 20180618 * 20180621 * 20180626 * 20180828

1.1 load data

[2]:

```
# load data

# load stim and spike data
dataset_name = "20180626"
dataset_filename = "data/{}.mat".format(dataset_name)

stim, spike_train, info = pysta.load_data(dataset_filename)

channel_names = [ch.replace("ch_","") for ch in info["channel_names"]]
# info["channel_names"]

# load cell type
cell_type = pd.read_csv("data/{}_cell_type.csv".format(dataset_name))
```

```
List of arrays in this file:
 <KeysViewHDF5 ['#refs#', 'channel_names', 'height', 'sampling_rate',
'spike_train', 'stim', 'width']>
Shape of the array stim:  (64, 9000)
Shape of the array spike_train:  (156, 9000)
length of the list channel_names:  156
sampling_rate:  10.0
```

1.2 result - eigenvalues

[3]:

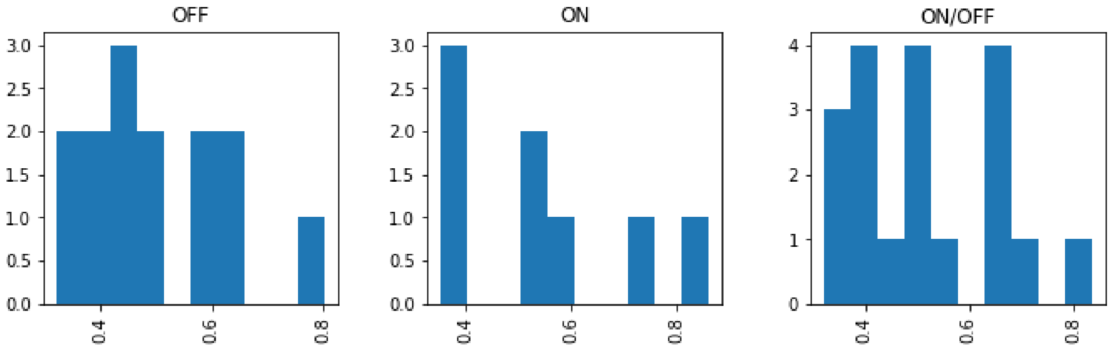
```
# read eigenvalus
all_eig_values = dict()
# eigen_values = list()
largest_eig_values = list()

folder_name = "{}_stc_tap8".format(dataset_name)
#folder_name = "stc_tap10_center_half"
for channel_name in channel_names: #info["channel_names"]:
    filename = "{}/ch_{}_eig_val.txt".format(folder_name,channel_name)
    eig_val = np.loadtxt(filename)

    all_eig_values[channel_name] = eig_val
#     eigen_values.append(eig_val)
    largest_eig_values.append(eig_val[0])

    #print(channel_name)
# plt.hist(largest_eig_values)

# all_eig_values
```

[4]:

```
# convert to DataFrame
eig = pd.DataFrame({"channel_name": channel_names, "largest_eig_values":
 ↪largest_eig_values})

results = cell_type.merge(eig, on="channel_name")
results.hist(column=["largest_eig_values"], by=["cell_type"], layout=(1,3),␣
 ↪figsize=(11,3))
```

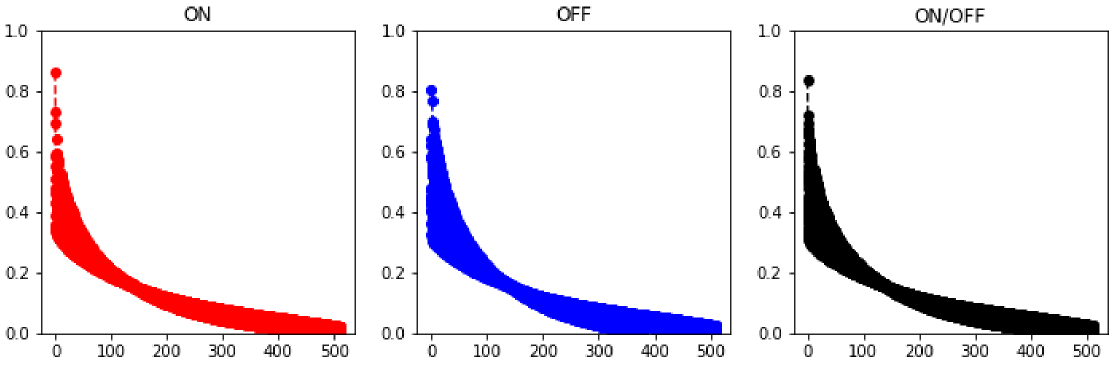
[4]:

```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x10a2a8fd0>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x1a1e3b2dd0>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x1a1e431b90>],
      dtype=object)
```



[5]:
```
# plot eigenvalues for cell type

plt.figure(figsize=(12,3.5))
ax=plt.subplot(131)
for channel_name in cell_type.loc[cell_type["cell_type"] ==␣
 ↪"ON"]["channel_name"]:
    #print(channel_name)
    plt.plot(all_eig_values[channel_name], 'or--')
ax.set_ylim(0, 1)
plt.title("ON")

ax=plt.subplot(132)
for channel_name in cell_type.loc[cell_type["cell_type"] ==␣
 ↪"OFF"]["channel_name"]:
    #print(channel_name)
    plt.plot(all_eig_values[channel_name], 'ob--')
ax.set_ylim(0, 1)
plt.title("OFF")

ax=plt.subplot(133)
for channel_name in cell_type.loc[cell_type["cell_type"] == "ON/
 ↪OFF"]["channel_name"]:
    #print(channel_name)
    plt.plot(all_eig_values[channel_name], 'ok--')
ax.set_ylim(0, 1)
plt.title("ON/OFF")
```

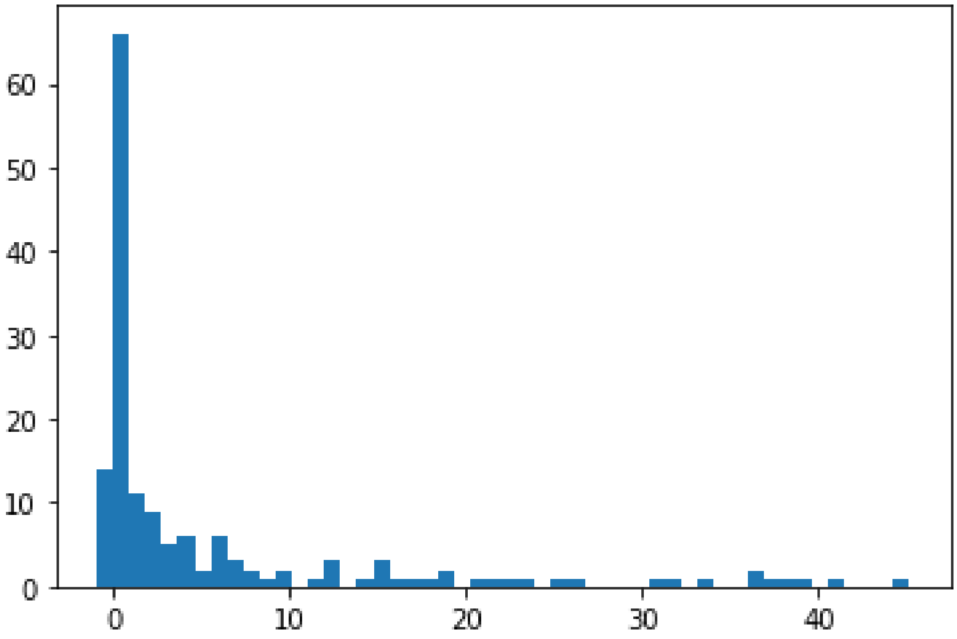
[5]:
```
Text(0.5, 1.0, 'ON/OFF')
```



1.3 result - kurtosis

[6]:
```
# load kurtosis
#tap = 5
Ks = np.loadtxt("{}/kurtosis.txt".format(folder_name))
plt.hist(Ks,50)

# store into a DataFrame
# remove "ch_" from channel names
kurtosis = pd.DataFrame({"channel_name": channel_names, "kurtosis": Ks})

kurtosis
```

[6]:
```
    channel_name  kurtosis
0            12a  0.107259
1            12b  0.679392
2            12c -0.497434
3            13a  0.078079
4            13b  6.185805
..           …         …
151          86c  2.206497
152          86d  6.641754
153          87a  0.064257
154          87b  0.051199
155          87c  4.326093

[156 rows x 2 columns]
```



[7]:
```
# merge with cell_type
#cell_type
results = results.merge(kurtosis, on="channel_name", how="outer")
#results = cell_type.merge(kurtosis, on="channel_name")
# results.hist(column=["kurtosis"], by=["cell_type"], layout=(1,3),␣
 ↪figsize=(12,3.5))
```

[8]:
```
results
```

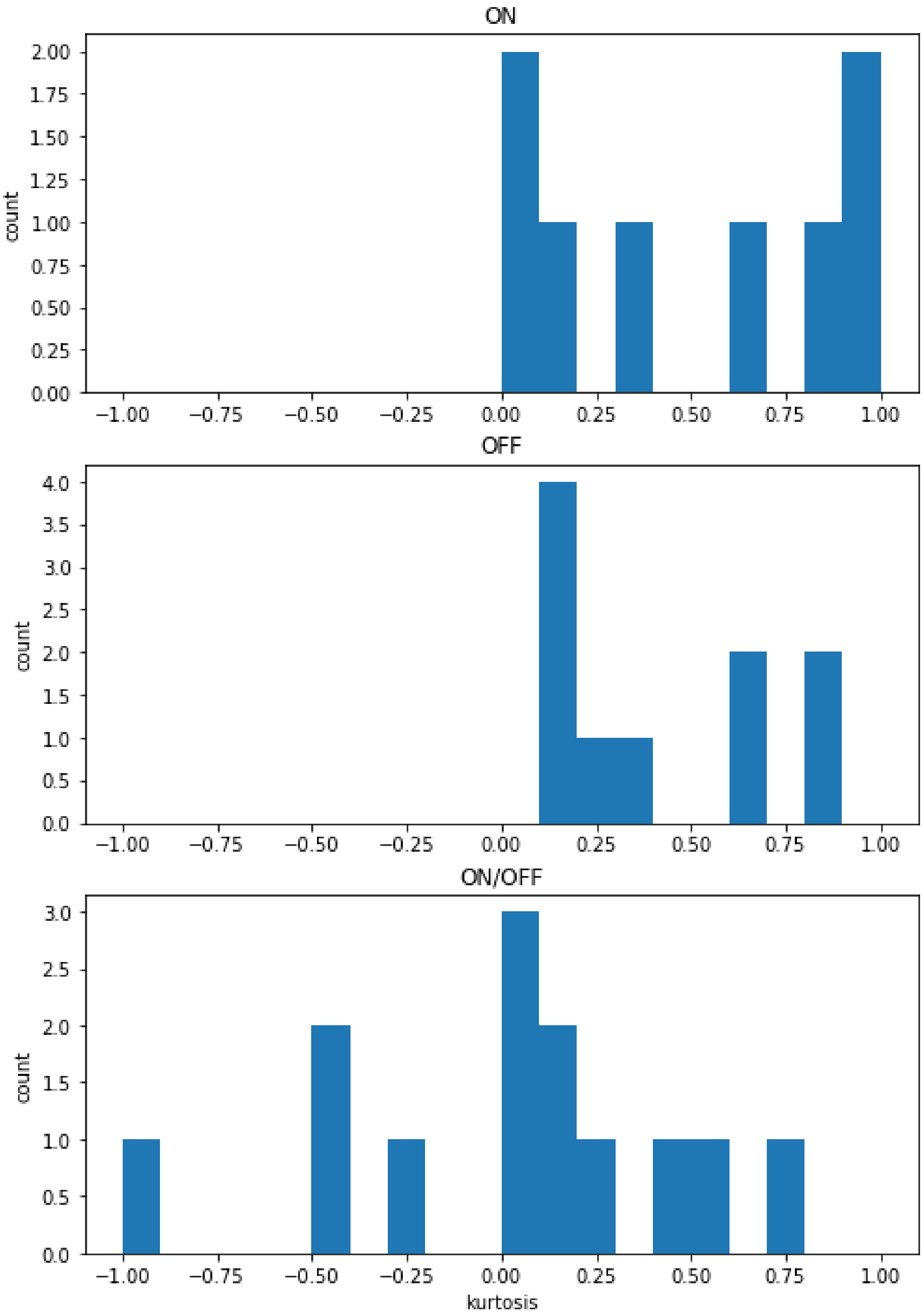
[8]:
```
    channel_name cell_type  largest_eig_values  kurtosis
0            12b       OFF            0.473982  0.679392
1            14a       OFF            0.617364  4.912770
2            14b       OFF            0.575853  4.484113
3            17a        ON            0.350985  0.080469
4            21a       OFF            0.402373  0.126339
..           …         …                  …         …
151          86b       NaN                 NaN  0.555748
152          86c       NaN                 NaN  2.206497
153          86d       NaN                 NaN  6.641754
154          87b       NaN                 NaN  0.051199
155          87c       NaN                 NaN  4.326093

[156 rows x 4 columns]
```

[9]:

```
k_on = results.loc[results["cell_type"]=="ON","kurtosis"]
k_off = results.loc[results["cell_type"]=="OFF","kurtosis"]
k_on_off = results.loc[results["cell_type"]=="ON/OFF","kurtosis"]

bins = np.linspace(-1,1,21)
# plt.hist(k_on, bins)
# plt.hist(k_off, bins)
# plt.hist(k_on_off, bins)

# plot separately
plt.figure(figsize=(8,12))
plt.subplot(3,1,1)
plt.hist(k_on, bins)
plt.title("ON")
# plt.xlabel("kurtosis")
plt.ylabel("count")

plt.subplot(3,1,2)
plt.hist(k_off, bins)
plt.title("OFF")
# plt.xlabel("kurtosis")
plt.ylabel("count")

plt.subplot(3,1,3)
plt.hist(k_on_off, bins)
plt.title("ON/OFF")
plt.xlabel("kurtosis")
plt.ylabel("count")
```

[9]:

```
Text(0, 0.5, 'count')
```


ON
count
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
OFF
count
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
ON/OFF
count
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
kurtosis

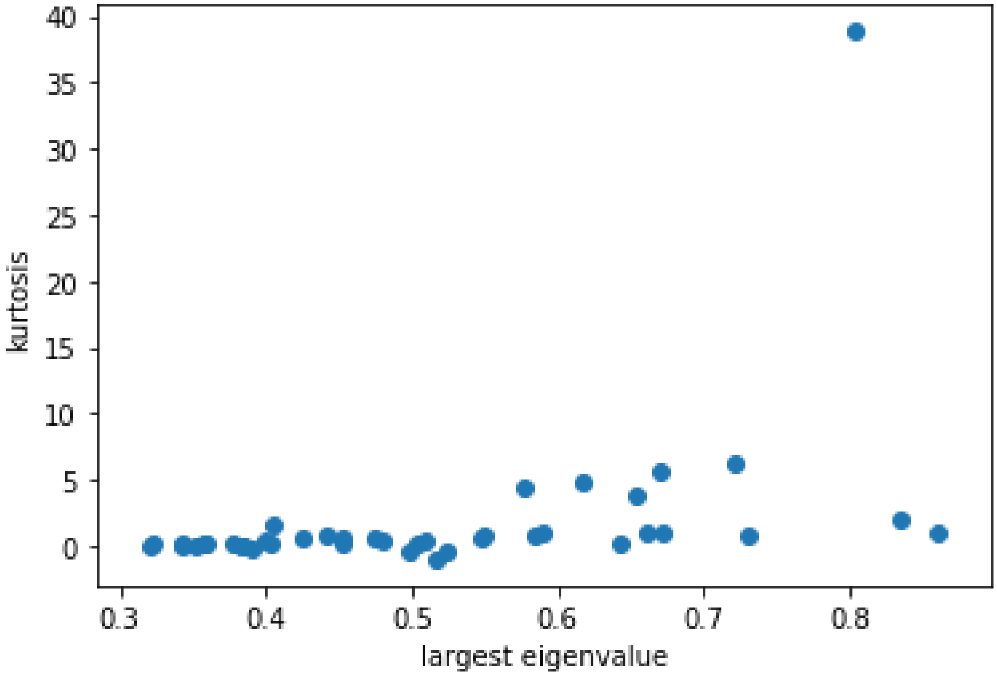
```
[10]: results.loc[results["kurtosis"]<0]
```

```
[10]:     channel_name cell_type  largest_eig_values  kurtosis
      5            22a    ON/OFF            0.523932 -0.493009
      6            22b    ON/OFF            0.498135 -0.465387
      18           36c    ON/OFF            0.515493 -0.974790
      19           37b    ON/OFF            0.390157 -0.261774
      42           12c       NaN                 NaN -0.497434
      47           13e       NaN                 NaN -0.066077
      77           33d       NaN                 NaN -0.189388
      78           35a       NaN                 NaN -0.003674
      80           35c       NaN                 NaN -0.951376
      83           35f       NaN                 NaN -0.057777
      84           35g       NaN                 NaN -0.387065
      90           37e       NaN                 NaN -0.960922
      93           41c       NaN                 NaN -0.244839
      104          46b       NaN                 NaN -0.079259
      109          48e       NaN                 NaN -0.734711
```

1.4 eigenvalues & kurtosis

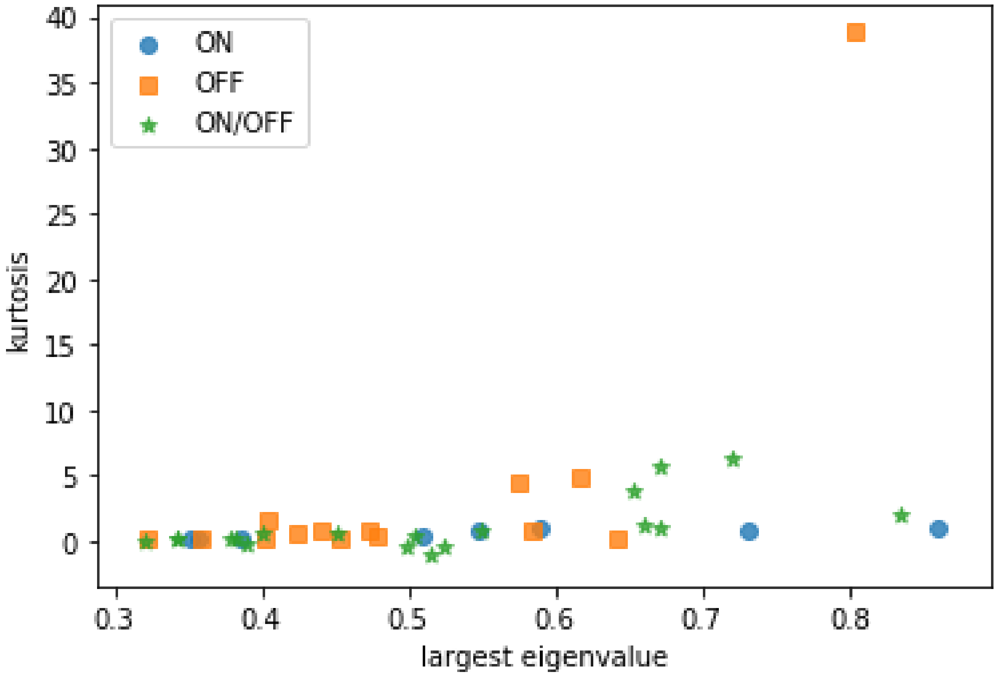
```
[11]: plt.scatter(results["largest_eig_values"], results["kurtosis"])
      plt.xlabel("largest eigenvalue")
      plt.ylabel("kurtosis")
```

```
[11]: Text(0, 0.5, 'kurtosis')
```



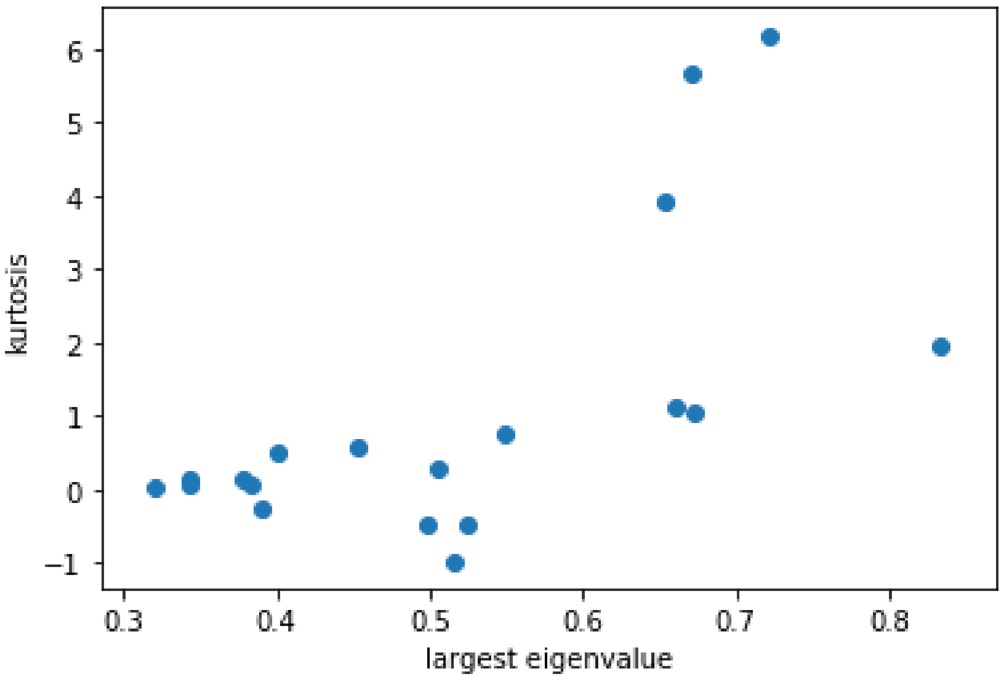
```
[12]: # plot for each cell type
      results_on = results.loc[results["cell_type"]=="ON"]
      results_off = results.loc[results["cell_type"]=="OFF"]
      results_on_off = results.loc[results["cell_type"]=="ON/OFF"]

      # plt.figure(figsize=(12,3))
      # plt.subplot(131)
      ax=plt.scatter(results_on["largest_eig_values"], results_on["kurtosis"],␣
       ↪marker="o", alpha=0.8)
      plt.xlabel("largest eigenvalue")
      plt.ylabel("kurtosis")

      # plt.subplot(132)
      plt.scatter(results_off["largest_eig_values"], results_off["kurtosis"],␣
       ↪marker="s", alpha=0.8)
      plt.xlabel("largest eigenvalue")
      plt.ylabel("kurtosis")

      # plt.subplot(133)
      plt.scatter(results_on_off["largest_eig_values"], results_on_off["kurtosis"],␣
       ↪marker="*", alpha=0.8)
      plt.xlabel("largest eigenvalue")
      plt.ylabel("kurtosis")

      plt.legend(["ON", "OFF", "ON/OFF"])
```

```
[12]: <matplotlib.legend.Legend at 0x1a1fa61310>
```



[13]:
```
plt.scatter(results_on_off["largest_eig_values"], results_on_off["kurtosis"],␣
 ↪marker="o")# , alpha=0.8)
plt.xlabel("largest eigenvalue")
plt.ylabel("kurtosis")

# plt.legend(["ON", "OFF", "ON/OFF"])
```

[13]:
```
Text(0, 0.5, 'kurtosis')
```

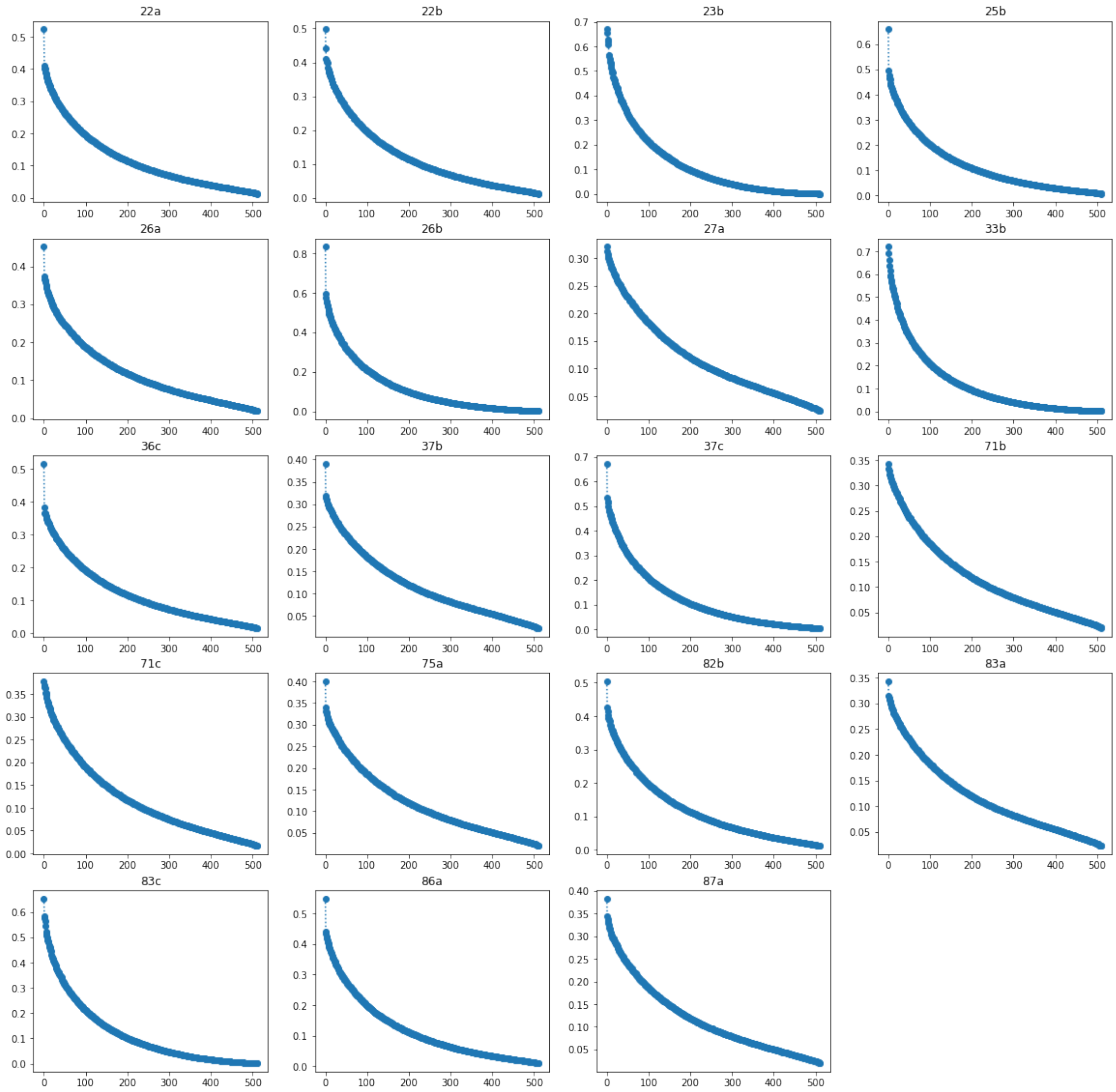


```
[14]: results_on_off
```

```
[14]:    channel_name cell_type  largest_eig_values  kurtosis
      5           22a    ON/OFF            0.523932 -0.493009
      6           22b    ON/OFF            0.498135 -0.465387
      7           23b    ON/OFF            0.670508  5.652101
      9           25b    ON/OFF            0.660372  1.105659
      10          26a    ON/OFF            0.452161  0.570836
      11          26b    ON/OFF            0.833974  1.940620
      12          27a    ON/OFF            0.320766  0.043030
      17          33b    ON/OFF            0.721018  6.185509
      18          36c    ON/OFF            0.515493 -0.974790
      19          37b    ON/OFF            0.390157 -0.261774
      20          37c    ON/OFF            0.672080  1.051079
      29          71b    ON/OFF            0.342335  0.066832
      30          71c    ON/OFF            0.378061  0.133544
      35          75a    ON/OFF            0.400070  0.497707
      36          82b    ON/OFF            0.504361  0.273066
      37          83a    ON/OFF            0.343108  0.146788
      38          83c    ON/OFF            0.653125  3.906088
      39          86a    ON/OFF            0.548798  0.765587
      40          87a    ON/OFF            0.382796  0.064257
```

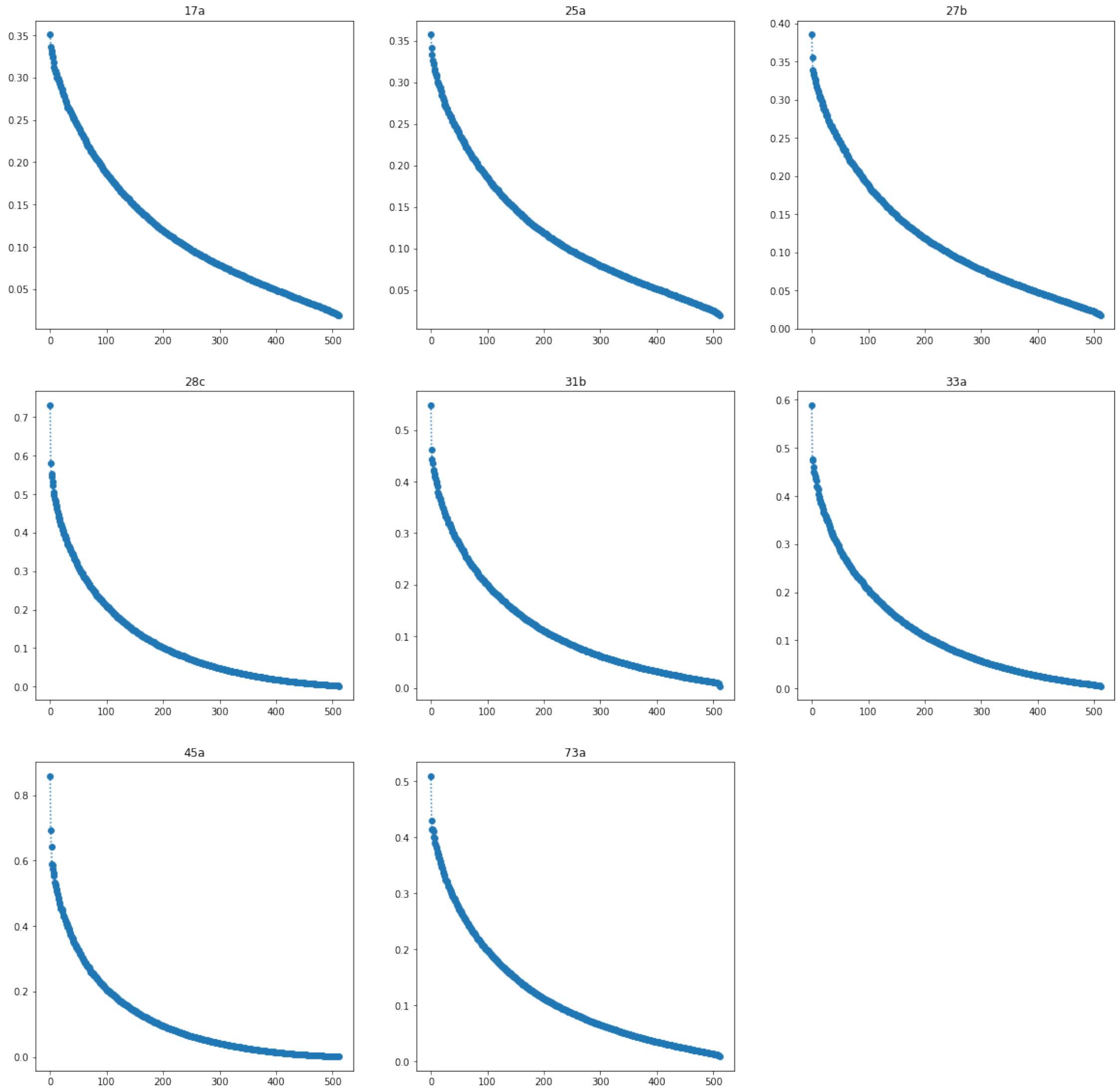
[15]:

```
# plot eigenvalues for ON/OFF cells

def plot_eigenvalues(all_eigen_values, channel_names):

    num_subplots=len(channel_names)
    num_row = int(np.ceil(np.sqrt(num_subplots)))
    num_col = int(np.ceil(num_subplots / num_row))

    plt.figure(figsize=(20,20))
    for i, channel_name in enumerate(channel_names):
        plt.subplot(num_row, num_col,i+1)
        plt.plot(all_eig_values[channel_name],"o:")
        plt.title(channel_name)

plot_eigenvalues(all_eig_values, results_on_off["channel_name"])
plt.savefig("{}/eigenvalues_on_off.png".format(folder_name))
```



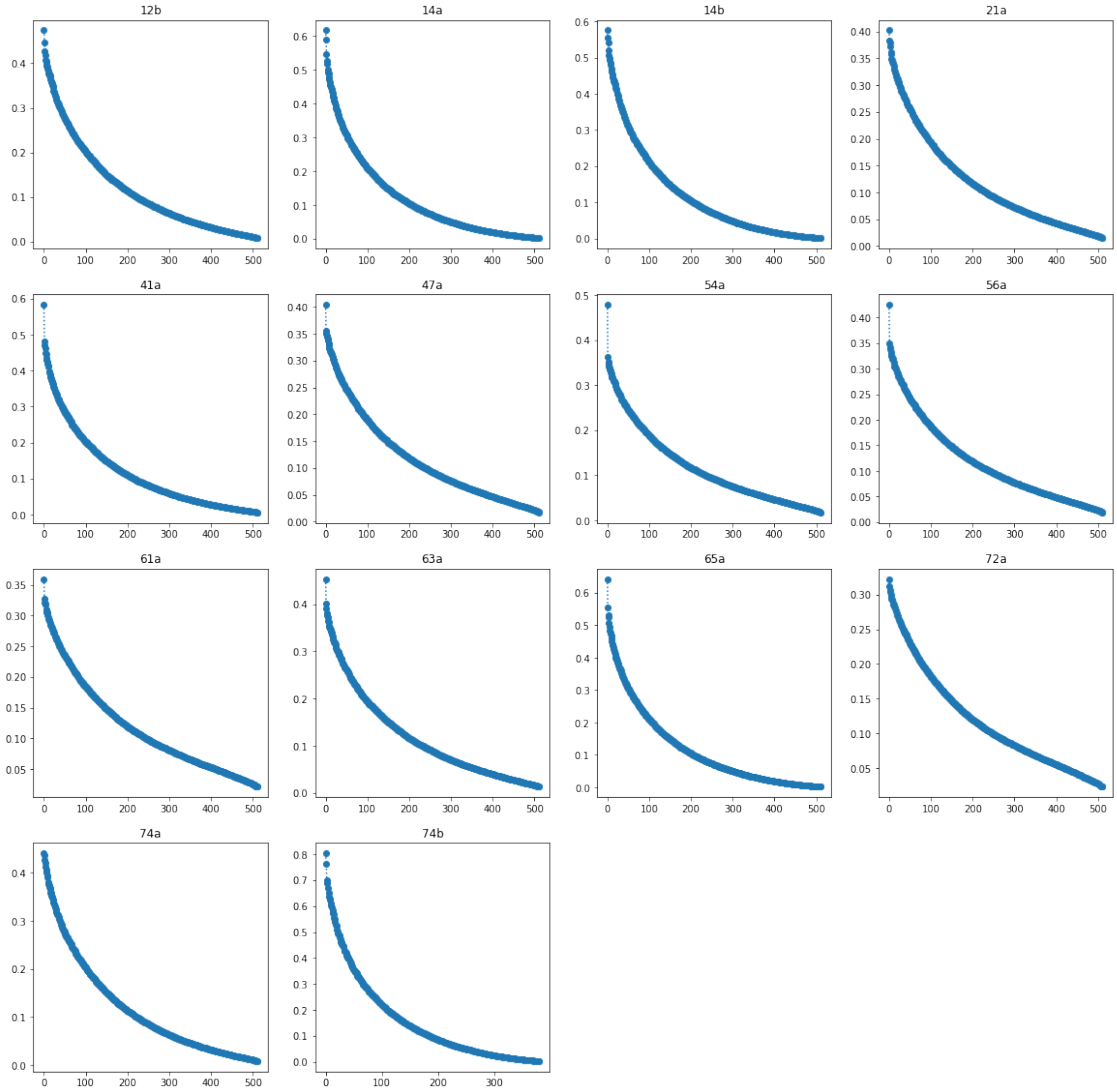
[16]:

```
plot_eigenvalues(all_eig_values, results_on["channel_name"])
plt.savefig("{}/eigenvalues_on.png".format(folder_name))
```



[17]:

```
plot_eigenvalues(all_eig_values, results_off["channel_name"])
plt.savefig("{}/eigenvalues_off.png".format(folder_name))
```


12b
0.0 0.1 0.2 0.3 0.4
0 100 200 300 400 500
14a
0.0 0.1 0.2 0.3 0.4 0.5 0.6
0 100 200 300 400 500
14b
0.0 0.1 0.2 0.3 0.4 0.5 0.6
0 100 200 300 400 500
21a
0.00 0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40
0 100 200 300 400 500
41a
0.0 0.1 0.2 0.3 0.4 0.5 0.6
0 100 200 300 400 500
47a
0.00 0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40
0 100 200 300 400 500
54a
0.0 0.1 0.2 0.3 0.4 0.5
0 100 200 300 400 500
56a
0.00 0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40
0 100 200 300 400 500
61a
0.05 0.10 0.15 0.20 0.25 0.30 0.35
0 100 200 300 400 500
63a
0.0 0.1 0.2 0.3 0.4
0 100 200 300 400 500
65a
0.0 0.1 0.2 0.3 0.4 0.5 0.6
0 100 200 300 400 500
72a
0.05 0.10 0.15 0.20 0.25 0.30
0 100 200 300 400 500
74a
0.0 0.1 0.2 0.3 0.4
0 100 200 300 400 500
74b
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8
0 100 200 300

```
[ ]:
```