rate data, as firing rate measures number of spikes within a timeframe. Faster firing rates directly indicate the neuronal excitability.

The previously mentioned study references the Kv4.2 and Kv4.3 channel subtypes (Simkin et al, 2015). In the human brain, there is high Kv4 protein expression in the CA2/CA3 pyramidal cells of the dentate gyrus (Birnbaum et al, 2004). Moreover, dendritic excitability is modulated by Kv4.3 channel through its rapid activation and inactivation of currents (Serôdio & Rudy, 1998). Therefore, we assume that neuronal hyperexcitability may be due to an increase in expression of these channels. The Allen BrainSpan atlas is a database that contains transcriptional information about human brain development. Kv4.3 is encoded by the KCDN3 gene and we can access the BrainSpan database to acquire data regarding human brain Kv4.3 expression. We will acquire KCDN3 gene expression levels in the hippocampal brain region and sort the data by subject age. This allows for the comparison of expression levels between older and younger age groups.

Overall, combining the Cell Types and BrainSpan database will allow us to see if hyperexcitability in aged neurons apply towards humans as well as towards rats and if differing Kv4.3 channel expression levels may be associated with excitability differences.

2.1 V. Hypothesis

If aged human hippocampal neurons show both greater excitability and Kv4.3 channel expression than young neurons, then this increased excitability is positively correlated with higher expression levels of KCND3 gene encoding for Kv4.3 channels.

As per mentioned by the Simkin et al. (2015) paper, the researchers found that aged CA3 hippocampal neurons in rats displayed hyperexcitability . They supposed that this hyperexcitability may be related to the overexpression of A-type K+ channels. Rats/mice are often used as subsitutes for human models, and we believe that Simkin et al.'s findings may translate over to human hippocampal neurons as well. This is supported by the fMRI blood-oxygen-level-dependent (BOLD) study also mentioend by Simkin et al. which showed healthy aged human subjects having an increase of fMRI BOLD activity in the CA3 region.

2.2 VI. Setup

[1]:
```
#packages that are essential for this project
import pandas as pd
import numpy as np

import requests
import json

import matplotlib.pyplot as plt
%matplotlib inline

from scipy import stats
```

2.3 VII. Data Wrangling

2.3.1 VII. 1. Getting electrophysiological features of human hippocampal cells

In this part, we will be working with dataset from Allen Cell Type website (https://celltypes.brain-map.org/). The Allen Software Development Kit (Allen SDK) is a source code for processing Allen Brain Atlas data, which includes the Cell Types Database. Electrophysiologcal features of human neurons can be found in the Allen Cell Types.

Ephys data can be downloaded from Allen Cell Types Database by importing the "Cell Types Cache" class, which will provide a Python interface for downloading data from the database. To access the database, we will then import CellTypesApi and initialize it as 'ctc'.

The 'get_cells' method downloads the metadata of all cells in the databse. We will filter for human cells by specifying the species as 'HUMAN' and assign the output to 'human_cells'.

Converting human cell metadata to a pandas dataframe allows us to see the data downloaded from the database in an organized manner. New dataframe will be organized by setting the index to 'id'.

[2]:
```
from allensdk.core.cell_types_cache import CellTypesCache
from allensdk.api.queries.cell_types_api import CellTypesApi

ctc = CellTypesCache(manifest_file='cell_types/manifest.json')
human_cells = ctc.get_cells(species=[CellTypesApi.HUMAN])

# setting up a dataframe that contains only human cell information
human_df = pd.DataFrame(human_cells)
human_df = human_df.set_index('id')
human_df.head()
```

[2]:
```
          reporter_status     cell_soma_location       species  \
id
525011903            None  [273.0, 354.0, 216.0]  Homo Sapiens
528642047            None    [69.0, 254.0, 96.0]  Homo Sapiens
537256313            None   [322.0, 255.0, 92.0]  Homo Sapiens
519832676            None    [79.0, 273.0, 91.0]  Homo Sapiens
596020931            None   [66.0, 220.0, 105.0]  Homo Sapiens

                            name structure_layer_name  structure_area_id  \
id
525011903     H16.03.003.01.14.02                    3              12113
528642047  H16.06.009.01.02.06.05                    5              12141
537256313     H16.03.006.01.05.02                    4              12141
519832676     H16.03.001.01.09.01                    3              12141
596020931     H17.06.009.11.04.02                    4              12141

          structure_area_abbrev transgenic_line dendrite_type     apical  \
id
525011903                 FroL                        spiny     intact
```

```
528642047                   MTG                        aspiny         NA
537256313                   MTG                         spiny  truncated
519832676                   MTG                         spiny  truncated
596020931                   MTG                        aspiny         NA

          reconstruction_type disease_state   donor_id structure_hemisphere  \
id
525011903                None      epilepsy  524848408                right
528642047                None      epilepsy  528574320                 left
537256313                None      epilepsy  536912860                right
519832676                full      epilepsy  518641172                 left
596020931                full         tumor  595954915                 left

           normalized_depth
id
525011903               NaN
528642047               NaN
537256313               NaN
519832676          0.290951
596020931          0.497825
```

Since our project is interested in hippocampal spiny neurons because they represent the neurons studied in Simkin et al. (2015), we will filter the dataframe to only contain spiny neurons in MTG.

[3]:
```
# filter data to contain only spiny neurons in MTG area
MTG_df = human_df[human_df['structure_area_abbrev']=='MTG']
MTG_spiny_df = MTG_df[MTG_df['dendrite_type']=='spiny']

MTG_spiny_df.head()
```

[3]:
```
          reporter_status    cell_soma_location       species  \
id
537256313            None  [322.0, 255.0, 92.0]  Homo Sapiens
519832676            None   [79.0, 273.0, 91.0]  Homo Sapiens
545608578            None  [312.0, 280.0, 89.0]  Homo Sapiens
561467633            None   [79.0, 273.0, 86.0]  Homo Sapiens
528706755            None  [70.0, 260.0, 108.0]  Homo Sapiens

                          name structure_layer_name  structure_area_id  \
id
537256313     H16.03.006.01.05.02                   4              12141
519832676     H16.03.001.01.09.01                   3              12141
545608578     H16.03.010.13.06.01                   3              12141
561467633     H16.06.013.12.08.05                   3              12141
528706755  H16.06.009.01.01.15.01                   2              12141

          structure_area_abbrev transgenic_line dendrite_type     apical  \
```

```
id
537256313                   MTG                          spiny  truncated
519832676                   MTG                          spiny  truncated
545608578                   MTG                          spiny     intact
561467633                   MTG                          spiny  truncated
528706755                   MTG                          spiny     intact

          reconstruction_type disease_state   donor_id structure_hemisphere  \
id
537256313                None      epilepsy  536912860                right
519832676                full      epilepsy  518641172                 left
545608578                None      epilepsy  545510854                right
561467633                None      epilepsy  561414332                 left
528706755       dendrite-only      epilepsy  528574320                 left

           normalized_depth
id
537256313               NaN
519832676          0.290951
545608578               NaN
561467633               NaN
528706755          0.134667
```

Now we need to get the electrophysiological properties data of the cells we're interested in. The 'get_ephys_feature()' method allows us to get the ephys data. And then we sort the dataframe by 'specimen_id'.

Since we are interested in data in both MTG_spiny_df and ephys_features, we are going to combine these two together by their same index ('specimen_id' is the same as 'id').

[4]:
```
# to get ephys data
ephys_features = pd.DataFrame(ctc.get_ephys_features()).set_index('specimen_id')
ephys_features.head()

# to combine both dataframes
MTGspiny_ephys_features = MTG_spiny_df.join(ephys_features)
print(MTGspiny_ephys_features.columns)
MTGspiny_ephys_features.head()
```

```
Index(['reporter_status', 'cell_soma_location', 'species', 'name',
       'structure_layer_name', 'structure_area_id', 'structure_area_abbrev',
       'transgenic_line', 'dendrite_type', 'apical', 'reconstruction_type',
       'disease_state', 'donor_id', 'structure_hemisphere', 'normalized_depth',
       'adaptation', 'avg_isi', 'electrode_0_pa', 'f_i_curve_slope',
       'fast_trough_t_long_square', 'fast_trough_t_ramp',
       'fast_trough_t_short_square', 'fast_trough_v_long_square',
       'fast_trough_v_ramp', 'fast_trough_v_short_square', 'has_burst',
       'has_delay', 'has_pause', 'id', 'input_resistance_mohm', 'latency',
```

```
       'peak_t_long_square', 'peak_t_ramp', 'peak_t_short_square',
       'peak_v_long_square', 'peak_v_ramp', 'peak_v_short_square',
       'rheobase_sweep_id', 'rheobase_sweep_number', 'ri', 'sag', 'seal_gohm',
       'slow_trough_t_long_square', 'slow_trough_t_ramp',
       'slow_trough_t_short_square', 'slow_trough_v_long_square',
       'slow_trough_v_ramp', 'slow_trough_v_short_square', 'tau',
       'threshold_i_long_square', 'threshold_i_ramp',
       'threshold_i_short_square', 'threshold_t_long_square',
       'threshold_t_ramp', 'threshold_t_short_square',
       'threshold_v_long_square', 'threshold_v_ramp',
       'threshold_v_short_square', 'thumbnail_sweep_id',
       'trough_t_long_square', 'trough_t_ramp', 'trough_t_short_square',
       'trough_v_long_square', 'trough_v_ramp', 'trough_v_short_square',
       'upstroke_downstroke_ratio_long_square',
       'upstroke_downstroke_ratio_ramp',
       'upstroke_downstroke_ratio_short_square', 'vm_for_sag', 'vrest'],
      dtype='object')
```

[4]:

id	reporter_status	cell_soma_location	species	\
537256313	None	[322.0, 255.0, 92.0]	Homo Sapiens	
519832676	None	[79.0, 273.0, 91.0]	Homo Sapiens	
545608578	None	[312.0, 280.0, 89.0]	Homo Sapiens	
561467633	None	[79.0, 273.0, 86.0]	Homo Sapiens	
528706755	None	[70.0, 260.0, 108.0]	Homo Sapiens	

id	name	structure_layer_name	structure_area_id	\
537256313	H16.03.006.01.05.02	4	12141	
519832676	H16.03.001.01.09.01	3	12141	
545608578	H16.03.010.13.06.01	3	12141	
561467633	H16.06.013.12.08.05	3	12141	
528706755	H16.06.009.01.01.15.01	2	12141	

id	structure_area_abbrev	transgenic_line	dendrite_type	apical	…	\
					…	
537256313	MTG		spiny	truncated	…	
519832676	MTG		spiny	truncated	…	
545608578	MTG		spiny	intact	…	
561467633	MTG		spiny	truncated	…	
528706755	MTG		spiny	intact	…	

id	trough_t_ramp	trough_t_short_square	trough_v_long_square	\
537256313	5.694547	1.389900	-52.125004	
519832676	9.962780	1.211020	-53.875004	
545608578	22.069340	1.112633	-54.343754	

```
561467633      6.479140              1.174147           -60.312504
528706755     10.173433              1.670800           -47.062500

           trough_v_ramp  trough_v_short_square  \
id
537256313    -51.520836             -72.900002
519832676    -52.416668             -73.693753
545608578    -54.968751             -75.156258
561467633    -57.989586             -71.020838
528706755    -52.302085             -72.343750

           upstroke_downstroke_ratio_long_square  \
id
537256313                               3.121182
519832676                               4.574865
545608578                               3.675430
561467633                               4.149998
528706755                               2.806181

           upstroke_downstroke_ratio_ramp  \
id
537256313                        3.464528
519832676                        3.817988
545608578                        3.665890
561467633                        3.667004
528706755                        3.007196

           upstroke_downstroke_ratio_short_square  vm_for_sag      vrest
id
537256313                                3.054681  -87.531250 -72.628105
519832676                                4.980603  -84.218758 -72.547661
545608578                                3.586321  -78.500000 -74.496262
561467633                                3.929834  -78.312500 -69.626610
528706755                                2.592416  -82.593758 -72.490135

[5 rows x 70 columns]
```

One of the ephys features that we want, average firing rate, could not be found in this dataset. Therefore, we will try to fetch the data online using RMA query functions.

[5]:
```
# RMA query to get average firing rate
service = "http://api.brain-map.org/api/v2/data/query.json?criteria="

specimen_id = list(MTGspiny_ephys_features.index)
avg_fr = []

for specimen in specimen_id:
```

```
    donor_result = requests.get("%smodel::ApiCellTypesSpecimenDetail, rma::
↪criteria, \
    specimen[id$eq%s]" % (service, specimen)).json()
    avg_fr.append(donor_result['msg'][0]['ef__avg_firing_rate'])


MTGspiny_ephys_features['avg_fr'] = avg_fr
MTGspiny_ephys_features
```

[5]:

id	reporter_status	cell_soma_location	species	…
537256313	None	[322.0, 255.0, 92.0]	Homo Sapiens	
519832676	None	[79.0, 273.0, 91.0]	Homo Sapiens	
545608578	None	[312.0, 280.0, 89.0]	Homo Sapiens	
561467633	None	[79.0, 273.0, 86.0]	Homo Sapiens	
528706755	None	[70.0, 260.0, 108.0]	Homo Sapiens	
…	…	…	…	
528636794	None	[69.0, 254.0, 96.0]	Homo Sapiens	
611823070	None	[333.0, 236.0, 122.0]	Homo Sapiens	
508298270	None	[325.0, 257.0, 102.0]	Homo Sapiens	
545612828	None	[312.0, 280.0, 89.0]	Homo Sapiens	
527952884	None	[67.0, 256.0, 110.0]	Homo Sapiens	

id	name	structure_layer_name	structure_area_id	…
537256313	H16.03.006.01.05.02	4	12141	
519832676	H16.03.001.01.09.01	3	12141	
545608578	H16.03.010.13.06.01	3	12141	
561467633	H16.06.013.12.08.05	3	12141	
528706755	H16.06.009.01.01.15.01	2	12141	
…	…	…	…	
528636794	H16.06.009.01.02.03.01	5	12141	
611823070	H17.06.012.14.10.02	4	12141	
508298270	H16.06.004.01.04.05	3	12141	
545612828	H16.03.010.13.05.03	3	12141	
527952884	H16.06.008.01.31.06	4	12141	

id	structure_area_abbrev	transgenic_line	dendrite_type	apical	…	\
					…	
537256313	MTG		spiny	truncated	…	
519832676	MTG		spiny	truncated	…	
545608578	MTG		spiny	intact	…	
561467633	MTG		spiny	truncated	…	
528706755	MTG		spiny	intact	…	
…	…	…	…	…	…	
528636794	MTG		spiny	truncated	…	
611823070	MTG		spiny	truncated	…	

```
508298270                   MTG                          spiny     intact  …
545612828                   MTG                          spiny     intact  …
527952884                   MTG                          spiny  truncated  …

          trough_t_short_square trough_v_long_square  trough_v_ramp  \
id
537256313              1.389900           -52.125004     -51.520836
519832676              1.211020           -53.875004     -52.416668
545608578              1.112633           -54.343754     -54.968751
561467633              1.174147           -60.312504     -57.989586
528706755              1.670800           -47.062500     -52.302085
…                         …                  …            …
528636794              1.669760           -54.531254     -56.062504
611823070              1.274645           -56.468750     -58.125002
508298270              1.025320           -46.562504     -51.989585
545612828              1.115453           -52.312500     -51.270837
527952884              1.525980           -55.781254     -56.895836

          trough_v_short_square  upstroke_downstroke_ratio_long_square  \
id
537256313           -72.900002                                3.121182
519832676           -73.693753                                4.574865
545608578           -75.156258                                3.675430
561467633           -71.020838                                4.149998
528706755           -72.343750                                2.806181
…                         …                                    …
528636794           -70.031258                                3.200447
611823070           -61.960942                                3.238006
508298270           -47.890627                                2.386624
545612828           -71.156253                                3.830058
527952884           -67.518752                                3.264279

           upstroke_downstroke_ratio_ramp  \
id
537256313                        3.464528
519832676                        3.817988
545608578                        3.665890
561467633                        3.667004
528706755                        3.007196
…                                   …
528636794                        3.496134
611823070                        3.082875
508298270                        2.421567
545612828                        3.942305
527952884                        3.302206

           upstroke_downstroke_ratio_short_square  vm_for_sag      vrest  \
```

```
id
537256313                                3.054681  -87.531250 -72.628105
519832676                                4.980603  -84.218758 -72.547661
545608578                                3.586321  -78.500000 -74.496262
561467633                                3.929834  -78.312500 -69.626610
528706755                                2.592416  -82.593758 -72.490135
…                                           …         …         …
528636794                                3.321839  -87.281258 -69.768448
611823070                                3.022499  -92.000008 -61.007416
508298270                                2.337120  -86.000000 -74.491547
545612828                                3.623513  -73.437508 -70.085281
527952884                                3.226462  -89.781258 -66.603539

              avg_fr
id
537256313  12.919897
519832676   4.066584
545608578        NaN
561467633  27.533040
528706755   5.936480
…                …
528636794  11.369744
611823070  14.107434
508298270  93.196645
545612828        NaN
527952884  12.739444

[236 rows x 71 columns]
```

We want a dataframe containing only all the informaton we want. All information will be stored into 'final_ephys_df' and sorted by 'donor_id'.

[6]:
```
final_ephys_df = MTGspiny_ephys_features[['id','donor_id', 'avg_isi', 'avg_fr']]
final_ephys_df = final_ephys_df.set_index('donor_id').sort_index()
final_ephys_df.head(15)
```

[6]:
```
                  id     avg_isi      avg_fr
donor_id
487502058  488401731   74.421250   13.437022
487502058  488393273  107.981875    9.260813
487502058  488412388  136.225833    7.340752
487502058  508991448         NaN         NaN
487502058  488419346  320.915000    3.116090
488771222  500861768   53.234118   18.784946
488771222  500861836  110.320000    9.064540
488771222  489390373   54.640000   18.301611
500830126  500990487  102.300000    9.775171
```

```
504919864  505690617  123.294286    8.110676
504919864  505693294  110.807500    9.024660
504919864  505691491  238.320000    4.196039
504921484  508312811   83.300000   12.004802
504921484  508279814    8.650000  115.606936
504921484  508394158   10.820000   92.421442
```

Lastly, we want to acquire the age of all the subject ids in our dataframe. Again, this information is not available in any of the dataframes above. To get the donor age, we will run an RMA query. We will also separate subjects into younger (0-20 years old) or older (21-40 years old) groups.

[7]:
```
#run an RMA query to get the human donor age info corresponding
#to the recordings on MTG spiny neurons
donor_id_ephys = final_ephys_df.index.values.tolist()
donor_age_yrs = []

for each_id in donor_id_ephys:
    donor_result_ephys = requests.get("%smodel::ApiCellTypesSpecimenDetail,\
        rma::criteria, specimen[donor_id$eq%s]" % (service, each_id)).json()

    each_age = donor_result_ephys['msg'][0]['donor__age']
    donor_age_yrs.append(int(each_age.split(' ')[0]))

# merge donor age info with the ephys dataframe that we have
final_ephys_df['donor_age_yrs'] = donor_age_yrs

# seperate donor age into two groups: younger and older.
older_donor = []
younger_donor = []
age_category = []

#sort out a age category by 0-20 years and 20-40 years
for age in donor_age_yrs:
    if age < 41:
        if age > 20:
            older_donor.append(age)
            age_category.append('older')
        else:
            younger_donor.append(age)
            age_category.append('younger')
    else:
        age_category.append ('out_of_range')

print(len(older_donor), len(younger_donor))

final_ephys_df['age_category'] = age_category
final_ephys_df
```

```
155 7
```

[7]:

donor_id	id	avg_isi	avg_fr	donor_age_yrs	age_category
487502058	488401731	74.421250	13.437022	65	out_of_range
487502058	488393273	107.981875	9.260813	65	out_of_range
487502058	488412388	136.225833	7.340752	65	out_of_range
487502058	508991448	NaN	NaN	65	out_of_range
487502058	488419346	320.915000	3.116090	65	out_of_range
…	…	…	…	…	…
611526465	611988589	175.060000	5.712327	23	older
611526465	611940594	70.884615	14.107434	23	older
611526465	611940196	42.649091	23.447159	23	older
614077275	614273989	174.635000	5.726229	30	older
614077275	614268309	319.506667	3.129825	30	older

[236 rows x 5 columns]

2.3.2 VII. 2. Getting the expression of voltage-gated potassium channel, Kv4.3, in human hippocampus across different age groups

In this section, we will interact with the Developing Human Transcriptome (https://www.brainspan.org/rnaseq/search/index.html) from Allen Brain Atlas to obtain potassium channel expression in hippocampus across human subjects of different ages. The gene code for Kv4.3 is KCND3 (Carasquillo et al. 2012).

Firstly, we need to find the gene id for the gene KCND3 by running an RMA query.

[8]:
```
service = "http://api.brain-map.org/api/v2/data/query.json?criteria="

# getting gene ID using the RMA query function
gene_acronym = 'KCND3'
probe_type = 'NcbiGene'

result = requests.get("%smodel::Gene,\
rma::criteria,[acronym$eq'%s\'][type$eq'%s\'],organism[name$eq'Homo Sapiens'],\
rma::options[only$eq'genes.id']" % (service, gene_acronym, probe_type)).json()

gene_id = result['msg'][1]['id']
probe_id = result['msg'][0]['id']

print ('The gene ID for KCND3 in the database is: ', gene_id)
```

```
The gene ID for KCND3 in the database is:  3727
```

Next, we want to get donor id and donor age information about the donors who provided the hippocampal KCND3 expression data, again by running RMA query. We will run a for loop to get a list of donor id and age.

[9]:
```
#Fetching data for all donors with KCND expression data at hippocampus
dev_human_info = requests.get("%sservice::dev_human_expression\
[set$eq'exon_microarray_genes'][probes$eq3727][structures$eq10294]" %␣
 ↪(service)).json()

#getting a list of donors, with their age donor id.
donor_info = dev_human_info['msg']['samples']

# examine the age of each donor, and categorize them into 'younger' or 'older'
younger_id = []
younger_age = []
older_id = []
older_age = []
category = []

for donor in donor_info:
    donor_age = donor['donor']['age']
    age_num = int(donor_age.split(' ')[0])

    # changing the format of all age to years
    if 'mos' in donor_age:
        age_num = round(age_num/12, 2)
        donor_age = str(age_num) + ' yrs'

    if not 'pcw' in donor_age:
        if 'yrs' in donor_age and age_num > 20:
            donor_age = age_num
            older_id.append(donor['donor']['id'])
            older_age.append(donor_age)
            category.append('older')
        else:
            donor_age = age_num
            younger_id.append(donor['donor']['id'])
            younger_age.append(donor_age)
            category.append('younger')

# add younger group and older group together as one large group
all_donors_id = younger_id + older_id
all_age = younger_age + older_age

print ('These are the IDs of all donors:', all_donors_id)
print ('\n')
print ('The following are their ages: ', all_age)
```

```
These are the IDs of all donors: [12296, 12890, 12830, 12979, 12980, 12841,
12981, 12289, 12831, 12984, 12832, 13057, 12300, 12290, 12302, 12303, 12304]
```

```
The following are their ages:  [0.33, 0.33, 1, 2, 3, 8, 8, 11, 13, 18, 19, 21,
23, 30, 36, 37, 40]
```

Then , we will get the KCND3 gene expression in all the subjects of interest. Gene expression is measured by exon microarray probes. We will run a for loop through donor_id to iterate out a list of expression levels.

[10]:
```
##Getting KCND3 expression from each donor
all_kv_exp_hippo = []

for each_donor in all_donors_id:
    each_kv_exp = requests.get("%sservice::
 ↪dev_human_expression[set$eq'exon_microarray_genes']\
    [probes$eq3727][structures$eq10294][donors$eq'%s\']" % (service,␣
 ↪each_donor)).json()

    # convert the default dict to a pd dataframe to get expression values
    each_kv_exp_msg = each_kv_exp['msg']
    gene_info_df = pd.DataFrame(each_kv_exp_msg['probes'])
    exp_lvls = gene_info_df['expression_level'].tolist()[0][0]

    if type (exp_lvls) == str:
        exp_lvls = float (exp_lvls)

    all_kv_exp_hippo.append (exp_lvls)

# combine expression values with donor IDs and ages
result_dict = {'donor_id': all_donors_id, 'donor_age': all_age,\
               'category': category, 'KCND3_expression': all_kv_exp_hippo}
result_df = pd.DataFrame (result_dict)
result_df
```

[10]:

	donor_id	donor_age	category	KCND3_expression
0	12296	0.33	younger	8.7525
1	12890	0.33	younger	NaN
2	12830	1.00	younger	8.8687
3	12979	2.00	younger	7.2277
4	12980	3.00	younger	7.1818
5	12841	8.00	younger	8.5025
6	12981	8.00	younger	NaN
7	12289	11.00	younger	NaN
8	12831	13.00	younger	8.4413
9	12984	18.00	younger	7.1473
10	12832	19.00	younger	NaN
11	13057	21.00	older	8.8564
12	12300	23.00	older	8.0415
13	12290	30.00	older	8.8200

```
14      12302       36.00     older             9.2727
15      12303       37.00     older             9.0706
16      12304       40.00     older             7.5593
```

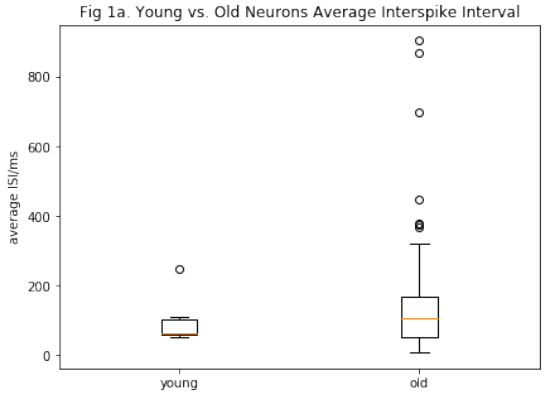
2.4 VIII. Data Visualization, Analysis & Results

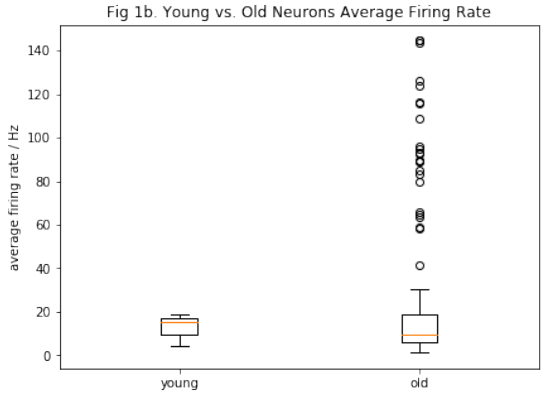
2.4.1 VIII. 1. Analyzing ephys data

We want to plot our data for average firing rate and average ISI against our old and young neuron groups. First, we need to filter our data to two different age groups, 'young_group' and 'old_group.' Then we will create dataframes that contain only average ISI and firing rate data.

[11]:
```
# plot several boxplots data against age category
fig, axes = plt.subplots(1, 2, figsize = (15, 5))

final_ephys_df = final_ephys_df.dropna()

young_group_ephys = final_ephys_df[final_ephys_df['age_category'] == 'younger']
old_group_ephys = final_ephys_df[final_ephys_df['age_category'] == 'older']

# plot for average isi
young_isi = young_group_ephys['avg_isi']
old_isi = old_group_ephys['avg_isi']

axes[0].boxplot([young_isi, old_isi])
axes[0].set_xticklabels(['young', 'old'])
axes[0].set_ylabel('average ISI/ms')
axes[0].set_title('Fig 1a. Young vs. Old Neurons Average Interspike Interval')

#plot for average firing rate
young_fr = young_group_ephys['avg_fr']
old_fr = old_group_ephys['avg_fr']

axes[1].boxplot([young_fr, old_fr])
axes[1].set_xticklabels(['young', 'old'])
axes[1].set_ylabel('average firing rate / Hz')
axes[1].set_title('Fig 1b. Young vs. Old Neurons Average Firing Rate')
```

[11]:
```
Text(0.5, 1.0, 'Fig 1b. Young vs. Old Neurons Average Firing Rate')
```

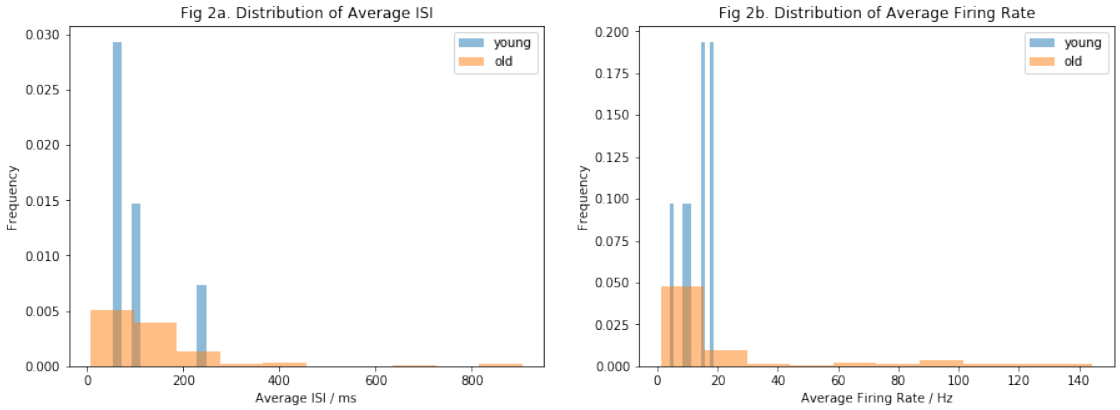




Now that we have our data plotted, we want to see if there is are significant differences of our data between young and old. Fig1 visusally does not give any significance between age groups since the boxplots have overlapping y values. To determine the significance statistically and which kind of ttest to perform, we want to perform skew test. However, since the sampling size is below 8, the skew test cannot give any results. Therefore, we decide to plot the distributions of the data and see from the distrbution if the data are skewed.

[12]:
```
# plot average isi distribution
fig, axes = plt.subplots(1, 2, figsize = (15, 5))

axes[0].hist(young_isi, alpha = 0.5,density = True)
axes[0].hist(old_isi, alpha = 0.5,density = True)
axes[0].legend(['young', 'old'])
axes[0].set_title('Fig 2a. Distribution of Average ISI')
axes[0].set_ylabel('Frequency')
axes[0].set_xlabel('Average ISI / ms')

# plot average firing rate distribution
axes[1].hist(young_fr, alpha = 0.5, density=True)
axes[1].hist(old_fr, alpha = 0.5, density=True)
axes[1].legend(['young', 'old'])
axes[1].set_title('Fig 2b. Distribution of Average Firing Rate')
axes[1].set_xlabel('Average Firing Rate / Hz')
axes[1].set_ylabel('Frequency')
```

[12]:
```
Text(0, 0.5, 'Frequency')
```



From the distributions above, we can see that all data are visibly skewed. Therefore, we will perform Mann-Whitney U t-test to see the significance of our data in two age groups. If the returned p-value is smaller than 0.05, the data has significance; and vice versa.

[13]:
```
# doing t-test to determine significance between age groups
stat_isi, p_value_isi = stats.mannwhitneyu(young_isi, old_isi)
stat_fr, p_value_fr = stats.mannwhitneyu(young_fr, old_fr)

if p_value_fr < 0.05 or p_value_isi < 0.05:
    print('P-value is smaller than 0.05, therefore the data is significant!')
else:
    print ('Both p-values are greater than 0.05. Therefore, neither the␣
 ↪inter-spike-interval nor average \nfiring rate across young and old subjects␣
 ↪is statistically significant.')
```

```
Both p-values are greater than 0.05. Therefore, neither the inter-spike-interval
nor average
firing rate across young and old subjects is statistically significant.
```

2.4.2 VIII. 2. Analyzing gene expression data

To visualize hippocampal expression of Kv4.3 in young and old subjects, we will create a boxplot of the expression values against age. First, we will drop all data with None type in order to create a boxplot. Then we will filter data according to age groups set dataframes with only gene expressions values inside.

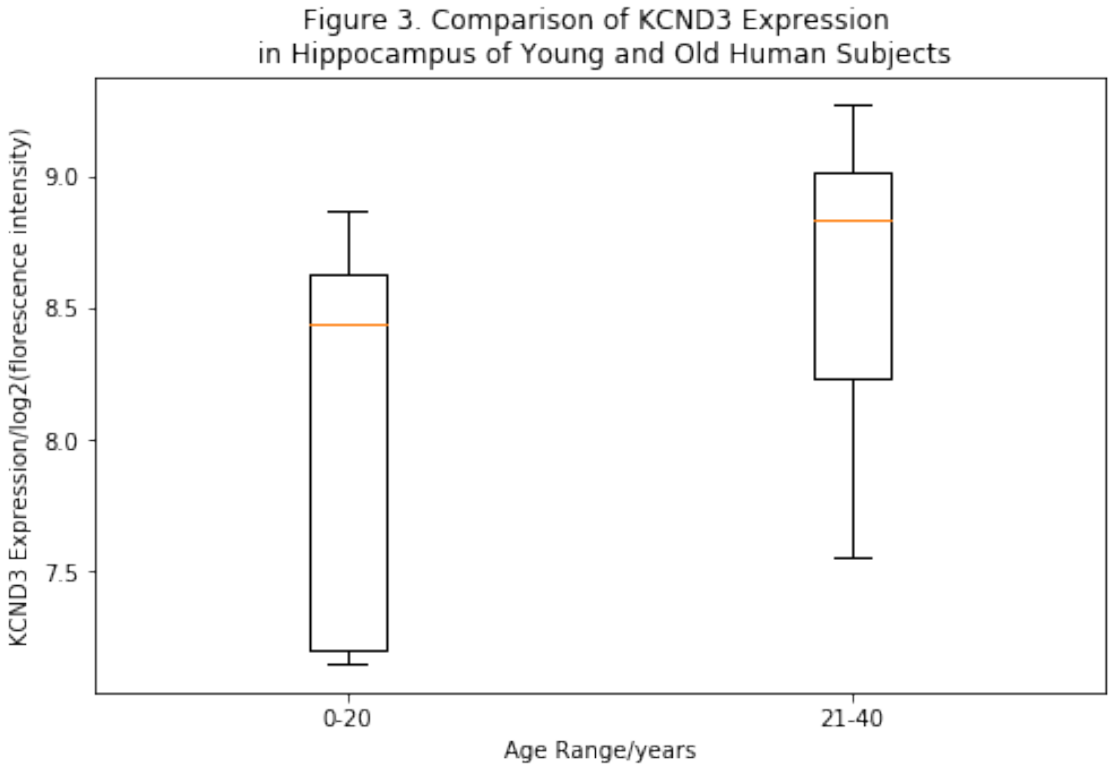
[14]:
```
# Dropping subjects without KCND3 expression
result_cleaned = result_df.dropna()
result_cleaned.set_index('donor_id')

# grouping the expression values to either 'young' or 'old' category
young_group_exp = result_cleaned[result_cleaned['category'] == 'younger']
old_group_exp = result_cleaned[result_cleaned['category'] == 'older']
```

```
young_exp = young_group_exp['KCND3_expression'].values.tolist()
old_exp = old_group_exp['KCND3_expression'].values.tolist()
print(len(young_group_exp), len(old_group_exp))

# convert each exp value in the list from string to float
young_exp = list(map(float, young_exp))
old_exp = list(map(float,old_exp))
exp_data = [young_exp, old_exp]
```

```
7 6
```

[15]:
```
# plotting KCDN3 expression against age groups
fig,ax = plt.subplots(figsize = (8,5))

ax.boxplot(exp_data)
ax.set_xlabel('Age Range/years')
ax.set_ylabel('KCND3 Expression/log2(florescence intensity)')
ax.set_xticklabels(['0-20','21-40'])
ax.set_title('Figure 3. Comparison of KCND3 Expression \n in Hippocampus of␣
↪Young and Old Human Subjects')
```

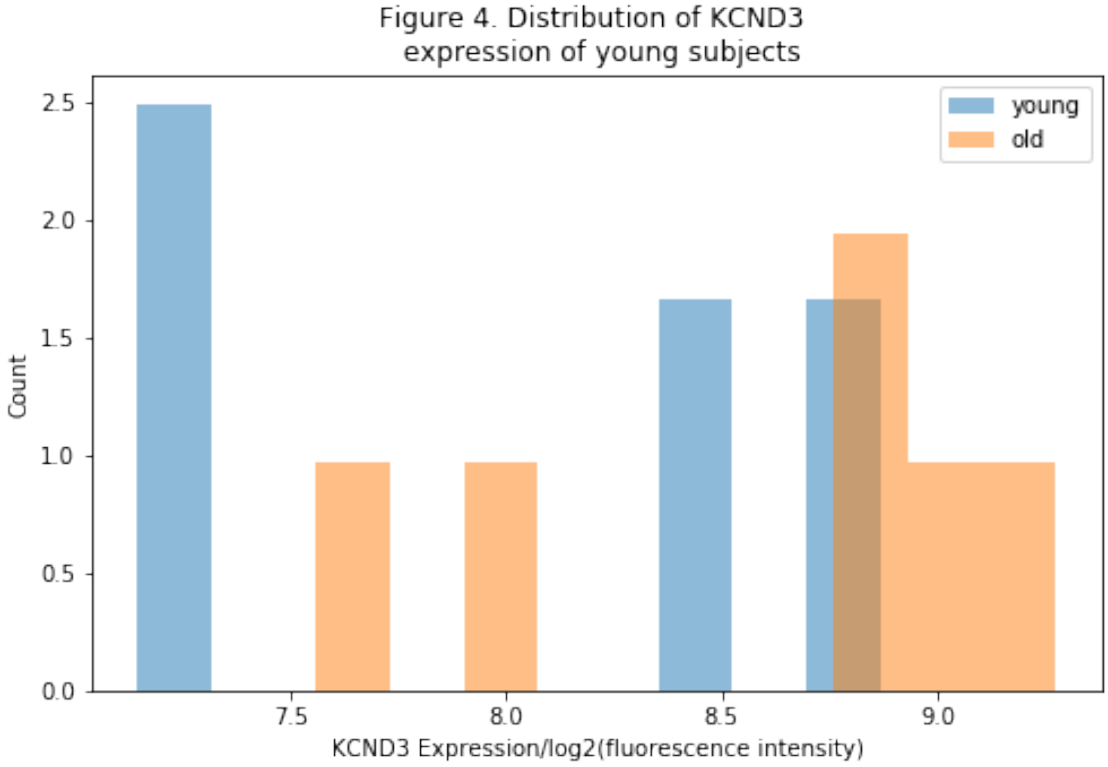
[15]:
```
Text(0.5, 1.0, 'Figure 3. Comparison of KCND3 Expression \n in Hippocampus of
Young and Old Human Subjects')
```



The above figure shows that younger subjects have a wider range of KCND3 expression in hippocampus than older subjects. But we need to validate the significance with t-test. Again, since the sampling size is below 8, so skewed test cannot be performed. To determine which t-test to use, we will visualize the distributions to see skewedness of the data.

[16]:
```
# plot KCND3 expression distribution
fig,ax = plt.subplots(figsize = (8,5))

ax.hist(young_exp, alpha = 0.5, density = True)
ax.hist(old_exp, alpha = 0.5, density = True)
ax.legend(['young', 'old'])
ax.set_xlabel('KCND3 Expression/log2(fluorescence intensity)')
ax.set_ylabel('Count')
ax.set_title('Figure 4. Distribution of KCND3 \n expression of young subjects')
```

[16]:
```
Text(0.5, 1.0, 'Figure 4. Distribution of KCND3 \n expression of young
subjects')
```



The above figure shows that expression values in both age groups are visibly skewed. From fig 3, younger subjects also seem to have lower hippocampal KCND3 expression compared to older subjects. In oder to figure out if such observation is statistically significant, we will conduct a

Mann-Whitney U Test against the two groups' expression values.

[17]:
```
# statistical test across young and old subjects
ttest_statistic, ttest_pVal = stats.mannwhitneyu(young_exp,old_exp)
print ("The p-Value for the t-test is: ", ttest_pVal)

if ttest_pVal > 0.05:
    print ('Since the p value is higher than 0.05, we have insufficient␣
 ↪evidence to claim that the hippocampal KCND expression of the young\
and old subjects are different.')
else:
    print('The p-value is lower than 0.05, therefore the gene expressions in␣
 ↪young and old groups are different!')
```

```
The p-Value for the t-test is:  0.06680720126885807
Since the p value is higher than 0.05, we have insufficient evidence to claim
that the hippocampal KCND expression of the youngand old subjects are different.
```

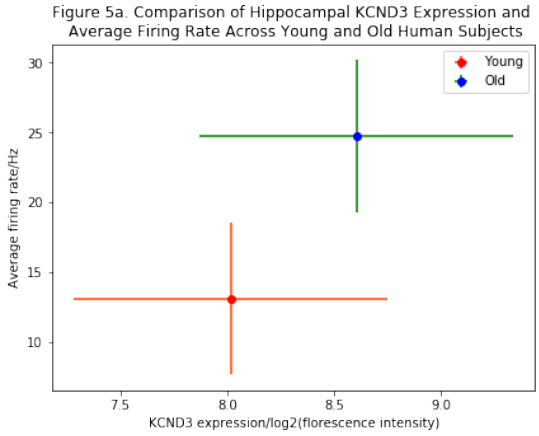
Lastly, we want to see if there is any correlation between neuronal excitability and gene expression. We want to concisely visualize, using an errobar scatterplot, the KCND3 gene expression, average ISI and average firing rate of hippocampal pyramidal neurons in young and old subjects. If the two scatters do not overlap, then we can conlude that KCND3 expression is correlated to neuronal excitability. If not, then we cannot conclude that KCND3 expression correlates to excitability.

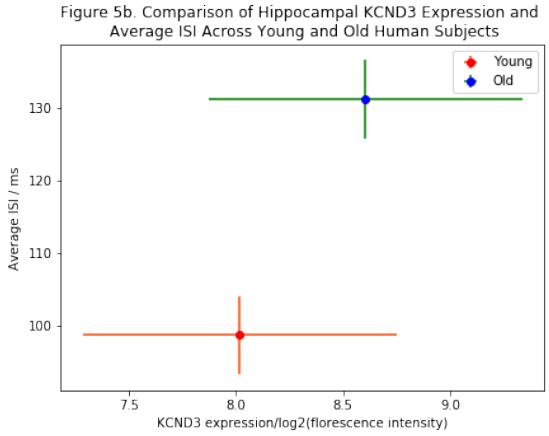
[25]:
```
fig,ax = plt.subplots(1, 2, figsize = (15,5))

# calculating mean and standard deviations for avg_isi and avg_fr
young_fr_average = young_fr.mean()
young_fr_std = young_fr.std()
old_fr_average = old_fr.mean()
old_fr_std = old_fr.std()

young_isi_avg = young_isi.mean()
young_isi_std = young_isi.std()
old_isi_avg = old_isi.mean()
old_isi_std = old_isi.std()

# calculating mean and standard deviations for young_exp and old_exp
young_exp_average = sum(young_exp)/len(young_exp)
young_exp_np = np.array(young_exp)
young_exp_std = young_exp_np.std()

old_exp_average = sum(old_exp)/len(old_exp)
old_exp_np = np.array(old_exp)
old_exp_std = old_exp_np.std()
```

```
print ('Figure 3 showed that there seemed to be little difference of the two␣
 ↪variables examined between the young and old age groups.' + '\nYoung␣
 ↪subjects have average hippoca\
mpal KCND3 expression of '+ str(young_exp_average) + '+/-' + str(young_exp_std)␣
 ↪+ ' log2(florescence intensity), and an average firing rate of ' +␣
 ↪str(young_fr_average) + '+/-' + str(young_fr_std) + ' ms. \
Old subjects have average hippoca\
mpal KCND3 expression of '+ str(old_exp_average) + '+/-' + str(old_exp_std) + '␣
 ↪log2(florescence intensity), and an average firing rate of ' +␣
 ↪str(old_fr_average) + '+/-' + str(old_fr_std) + ' ms. \nThe standard␣
 ↪deviations were large compared to differences in the average values.')

# setting up x and y axes
x1 = [young_exp_average]
y1 = [young_fr_average]
x2 = [old_exp_average]
y2 = [old_fr_average]

x3 = [young_exp_average]
y3 = [young_isi_avg]
x4 = [old_exp_average]
y4 = [old_isi_avg]

x_errorbar = [young_exp_std,old_exp_std]
y_errorbar = [young_fr_std, old_fr_std]

# plotting an errorbar scatterplot with all our data
ax[0].errorbar(x1,y1,xerr = x_errorbar[0], yerr = y_errorbar[0],fmt = 'o',␣
 ↪color = 'red', ecolor = 'orangered')
ax[0].errorbar(x2,y2,xerr = x_errorbar[0], yerr = y_errorbar[0],fmt = 'o',color␣
 ↪= 'blue', ecolor = 'green')
ax[0].legend (['Young','Old'])
ax[0].set_xlabel('KCND3 expression/log2(florescence intensity)')
ax[0].set_ylabel('Average firing rate/Hz')
ax[0].set_title('Figure 5a. Comparison of Hippocampal KCND3 Expression and \n␣
 ↪Average Firing Rate Across Young and Old Human Subjects')

ax[1].errorbar(x3,y3,xerr = x_errorbar[0], yerr = y_errorbar[0],fmt = 'o',␣
 ↪color = 'red', ecolor = 'orangered')
ax[1].errorbar(x4,y4,xerr = x_errorbar[0], yerr = y_errorbar[0],fmt = 'o',color␣
 ↪= 'blue', ecolor = 'green')
ax[1].legend (['Young','Old'])
ax[1].set_xlabel('KCND3 expression/log2(florescence intensity)')
ax[1].set_ylabel('Average ISI / ms')
ax[1].set_title('Figure 5b. Comparison of Hippocampal KCND3 Expression and \n␣
 ↪Average ISI Across Young and Old Human Subjects')
```

```
Figure 3 showed that there seemed to be little difference of the two variables
examined between the young and old age groups.
Young subjects have average hippocampal KCND3 expression of
8.0174+/-0.7328052148130106 log2(florescence intensity), and an average firing
rate of 13.11523237573175+/-5.43463184379038 ms. Old subjects have average
hippocampal KCND3 expression of 8.603416666666666+/-0.6031914163200787
log2(florescence intensity), and an average firing rate of
24.754531564414783+/-35.06204134820079 ms.
The standard deviations were large compared to differences in the average
values.
```

[25]:
```
Text(0.5, 1.0, 'Figure 5b. Comparison of Hippocampal KCND3 Expression and \n
 Average ISI Across Young and Old Human Subjects')
```





3 IX. Discussion & Conclusion

We saw that there is no statistical difference of Kv4.3 expression, average firing rate or average ISI examined in hippocampal cells between younger (0-20 years old) human subjects and older (21-40 years old) subjects. Therefore, we conclude that hippocampal neurons in older adults do not have increased excitability or KCND3 gene expression compared to those of young adults. However, according to fig5 which shows the errors of gene expression against either average firing rate and ISI, there are no overlaps between the scatterplots. Therefore, we can conclude from fig5 that neuronal excitability could correlate with KCND3 gene expression. This result suggest that neuronal excitability could be mediated by Kv4.3 expression. Nonetheless, we cannot conclude that either gene expression or neuronal excitability is correlated to age.

Several limitations exist for our project. First of all, there was no data precisely for 'pyramidal neurons in the hippocampus' in the two datasets we worked with. Regarding potassium channel expression, we were only able to look at the expression data for hippocampus in general. It is known, however, that there are over 47 types of cells in the hippocampus (Zeisel et al. 2015), meaning that general hippocampal KCND3 expression may not accurately represent the expression