Meeting 4

Arjun Sridhar

09/25/2020

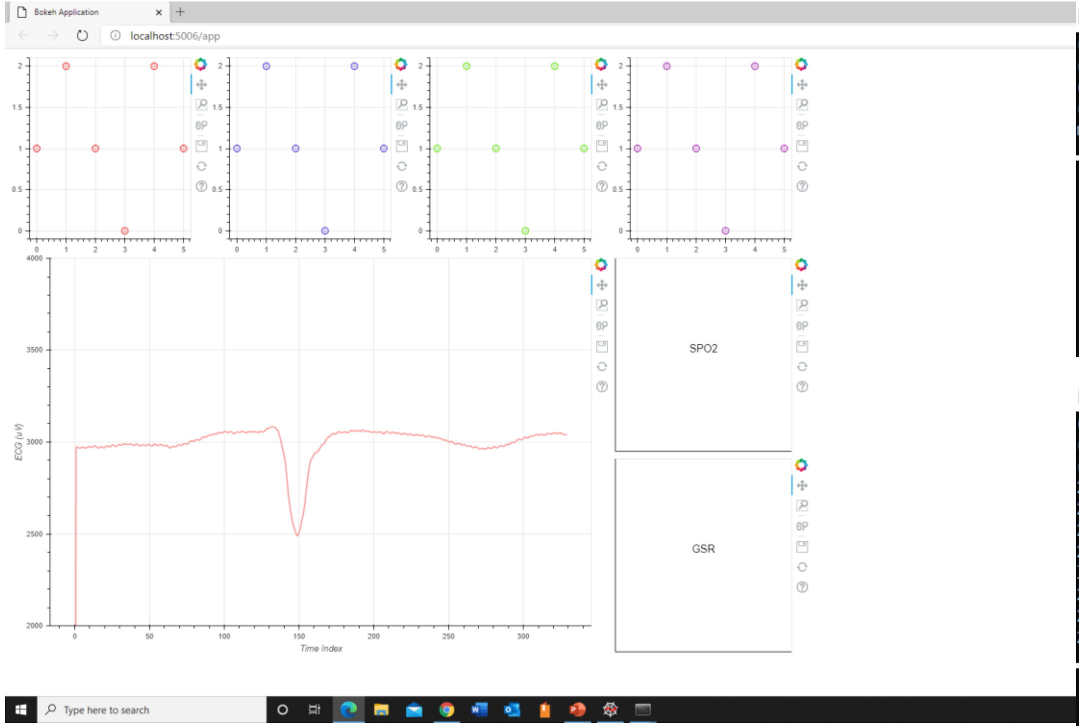
Objectives

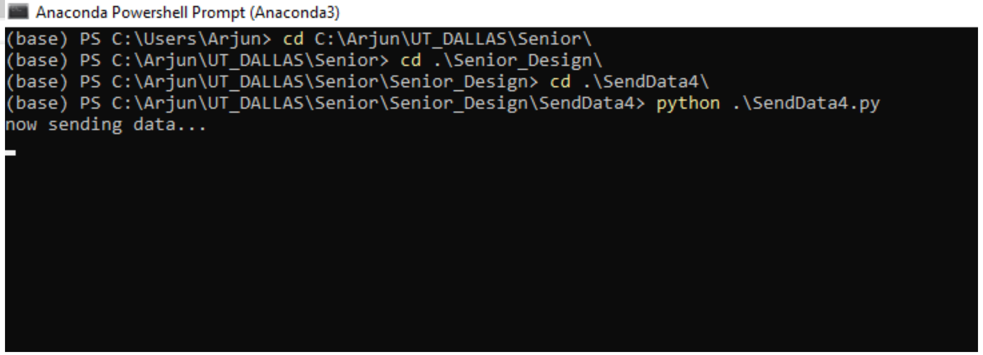
- Get app bones running

- Power Spectrum Function

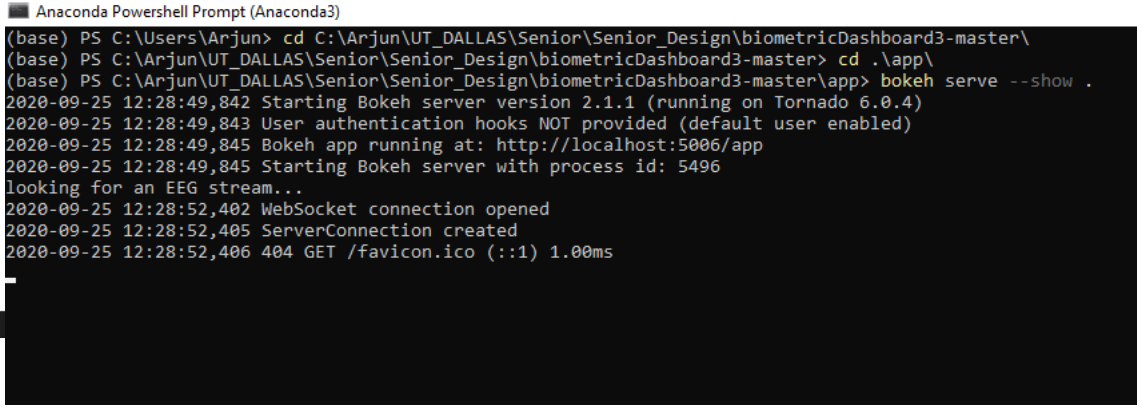
- Implement static EEG visualization using MNE

Outcomes



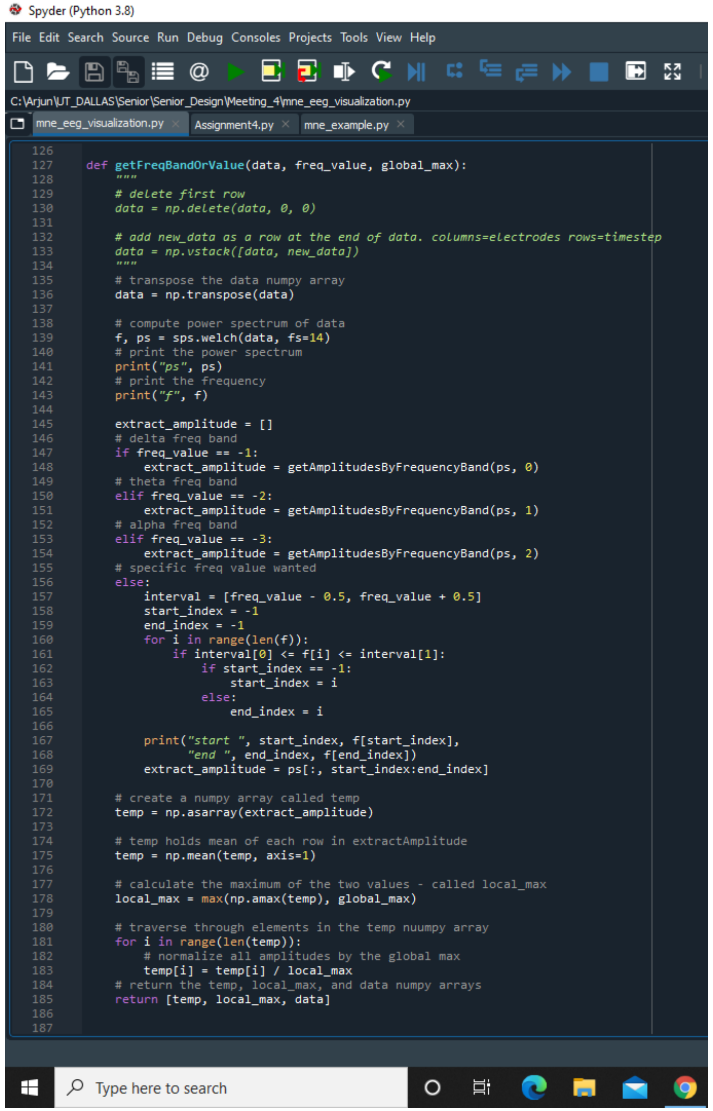


```
(base) PS C:\Users\Arjun> cd C:\Arjun\UT_DALLAS\Senior\
(base) PS C:\Arjun\UT_DALLAS\Senior> cd .\Senior_Design\
(base) PS C:\Arjun\UT_DALLAS\Senior\Senior_Design> cd .\SendData4\
(base) PS C:\Arjun\UT_DALLAS\Senior\Senior_Design\SendData4> python .\SendData4.py
now sending data...
```

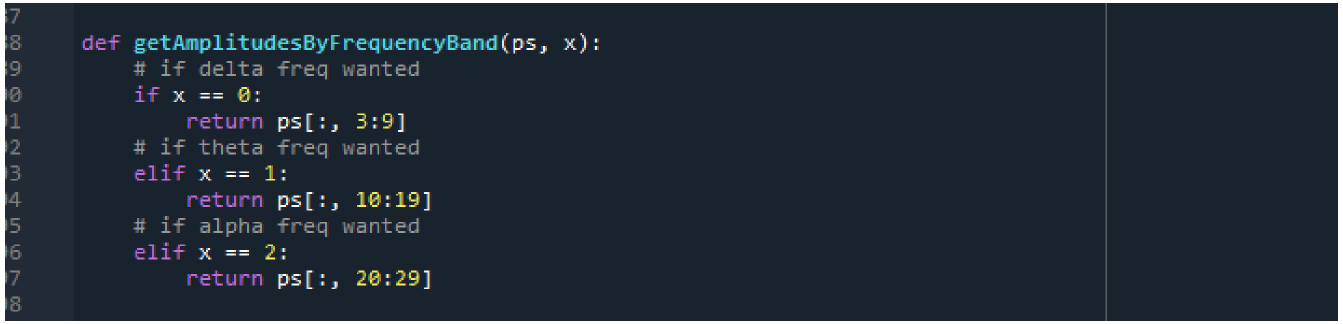


```
(base) PS C:\Users\Arjun> cd C:\Arjun\UT_DALLAS\Senior\Senior_Design\biometricDashboard3-master\
(base) PS C:\Arjun\UT_DALLAS\Senior\Senior_Design\biometricDashboard3-master> cd .\app\
(base) PS C:\Arjun\UT_DALLAS\Senior\Senior_Design\biometricDashboard3-master\app> bokeh serve --show .
2020-09-25 12:28:49,842 Starting Bokeh server version 2.1.1 (running on Tornado 6.0.4)
2020-09-25 12:28:49,843 User authentication hooks NOT provided (default user enabled)
2020-09-25 12:28:49,845 Bokeh app running at: http://localhost:5006/app
2020-09-25 12:28:49,845 Starting Bokeh server with process id: 5496
looking for an EEG stream...
2020-09-25 12:28:52,402 WebSocket connection opened
2020-09-25 12:28:52,405 ServerConnection created
2020-09-25 12:28:52,406 404 GET /favicon.ico (::1) 1.00ms
```

Outcomes

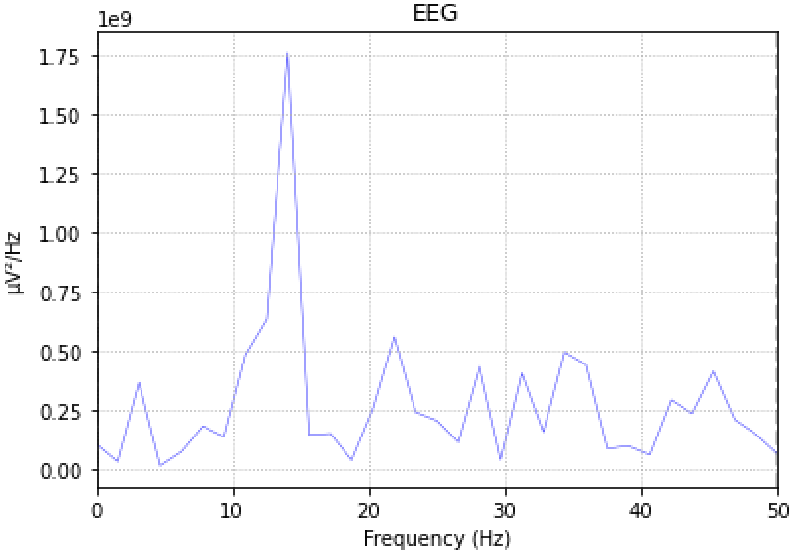

Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Arjun\UT_DALLAS\Senior\Senior_Design\Meeting_4\mne_eeg_visualization.py
mne_eeg_visualization.py
Assignment4.py
mne_example.py
Type here to search

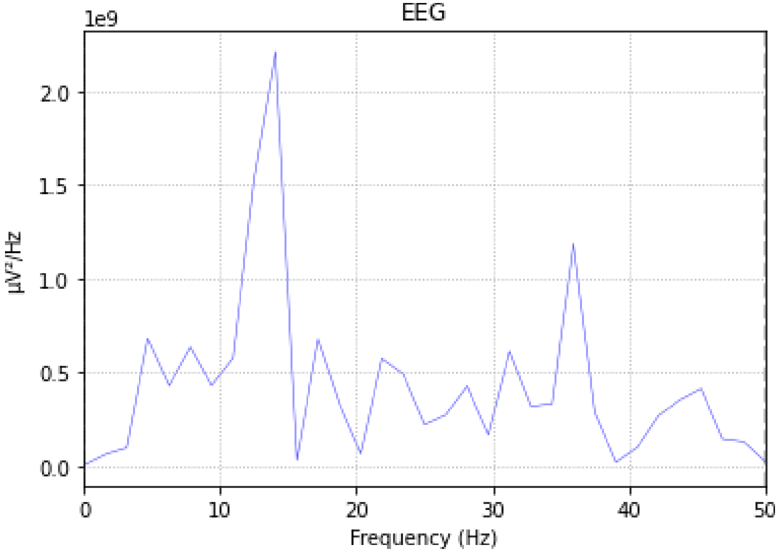
```python

def getFreqBandOrValue(data, freq_value, global_max):
    """
    # delete first row
    data = np.delete(data, 0, 0)

    # add new_data as a row at the end of data. columns=electrodes rows=timestep
    data = np.vstack([data, new_data])
    """
    # transpose the data numpy array
    data = np.transpose(data)

    # compute power spectrum of data
    f, ps = sps.welch(data, fs=14)
    # print the power spectrum
    print("ps", ps)
    # print the frequency
    print("f", f)

    extract_amplitude = []
    # delta freq band
    if freq_value == -1:
        extract_amplitude = getAmplitudesByFrequencyBand(ps, 0)
    # theta freq band
    elif freq_value == -2:
        extract_amplitude = getAmplitudesByFrequencyBand(ps, 1)
    # alpha freq band
    elif freq_value == -3:
        extract_amplitude = getAmplitudesByFrequencyBand(ps, 2)
    # specific freq value wanted
    else:
        interval = [freq_value - 0.5, freq_value + 0.5]
        start_index = -1
        end_index = -1
        for i in range(len(f)):
            if interval[0] <= f[i] <= interval[1]:
                if start_index == -1:
                    start_index = i
                else:
                    end_index = i

        print("start ", start_index, f[start_index],
              "end ", end_index, f[end_index])
        extract_amplitude = ps[:, start_index:end_index]

    # create a numpy array called temp
    temp = np.asarray(extract_amplitude)

    # temp holds mean of each row in extractAmplitude
    temp = np.mean(temp, axis=1)

    # calculate the maximum of the two values - called local_max
    local_max = max(np.amax(temp), global_max)

    # traverse through elements in the temp nuumpy array
    for i in range(len(temp)):
        # normalize all amplitudes by the global max
        temp[i] = temp[i] / local_max
    # return the temp, local_max, and data numpy arrays
    return [temp, local_max, data]

```

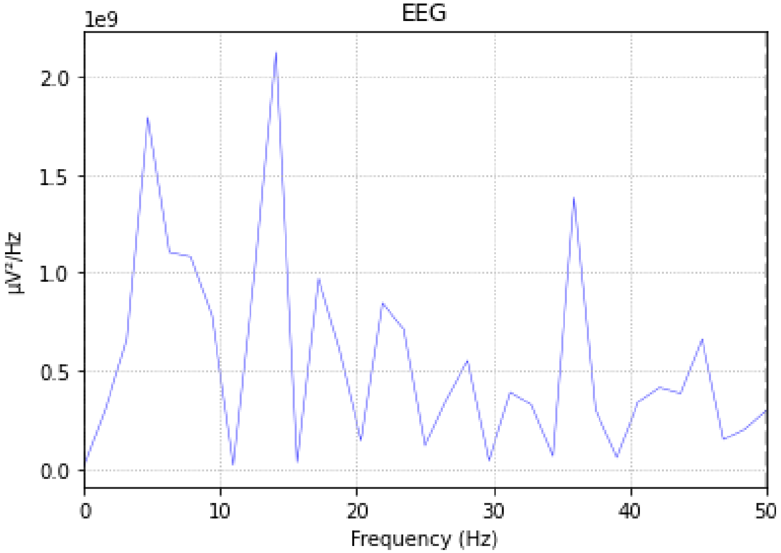


```python

def getAmplitudesByFrequencyBand(ps, x):
    # if delta freq wanted
    if x == 0:
        return ps[:, 3:9]
    # if theta freq wanted
    elif x == 1:
        return ps[:, 10:19]
    # if alpha freq wanted
    elif x == 2:
        return ps[:, 20:29]

```

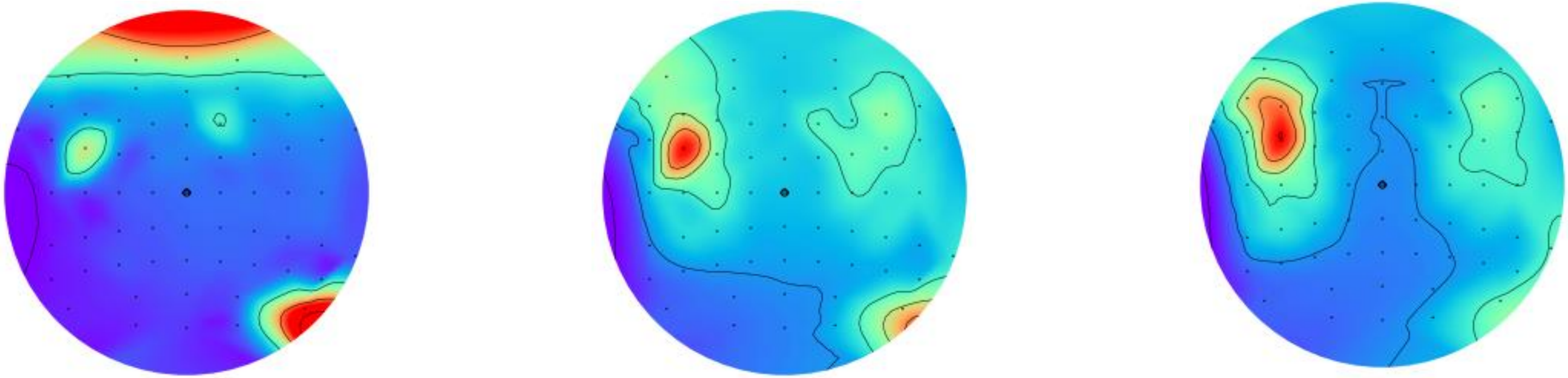
Outcomes


EEG
1e9
µV²/Hz
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
10
20
30
40
50
Frequency (Hz)


EEG
1e9
µV²/Hz
2.0
1.5
1.0
0.5
0.0
0
10
20
30
40
50
Frequency (Hz)


EEG
1e9
µV²/Hz
2.0
1.5
1.0
0.5
0.0
0
10
20
30
40
50
Frequency (Hz)

Outcomes



Looking Forward

- Fix issues in visualization and add improvements

- Add new data