Perceive Medtronic-CU Anschutz Collaboration

LEFT

Part 1: Import all necessary values and create global variables.

```
%% loads in json file
%loads in file
cd('C:\Users\sydne\Documents\github\perceive\patientData\Patient1_0524')
jsonFiles = 'Report_Json_Session_Report_Patient1_right_init.json';
js = jsondecode(fileread(jsonFiles));
%sets it back to the path where all the other functions are
cd('C:\Users\sydne\Documents\github\perceive')

%declares "global" variables AND determines if .json comes from
%single or double battery B)
channels = unique({js.LfpMontageTimeDomain.Channel}, 'stable');
leng = numel({js.LfpMontageTimeDomain.Channel});
startindex = 7; %MUST BE MANUALLY SET IN CLINIC
sides = {'LEFT', 'RIGHT'};
if any((contains(channels, sides{1}))) && any((contains(channels, sides{2})))
    doubleBattery = false;
    %json has both left and right data
    channelsLeft = channels(contains(channels, sides{1}));
    channelsRight = channels(contains(channels, sides{2}));
    channelSides = {channelsLeft, channelsRight};
    highestBetas = {' ', ' '};
else
    doubleBattery = true;
    %added this to help with titling figures
    if contains(channels, sides{1})
        side = 'LEFT';
    else
        side = 'RIGHT';
    end
end
```

Part 2: Create freq vs pwr plot for all 6 contacts. Also reports the max betas.

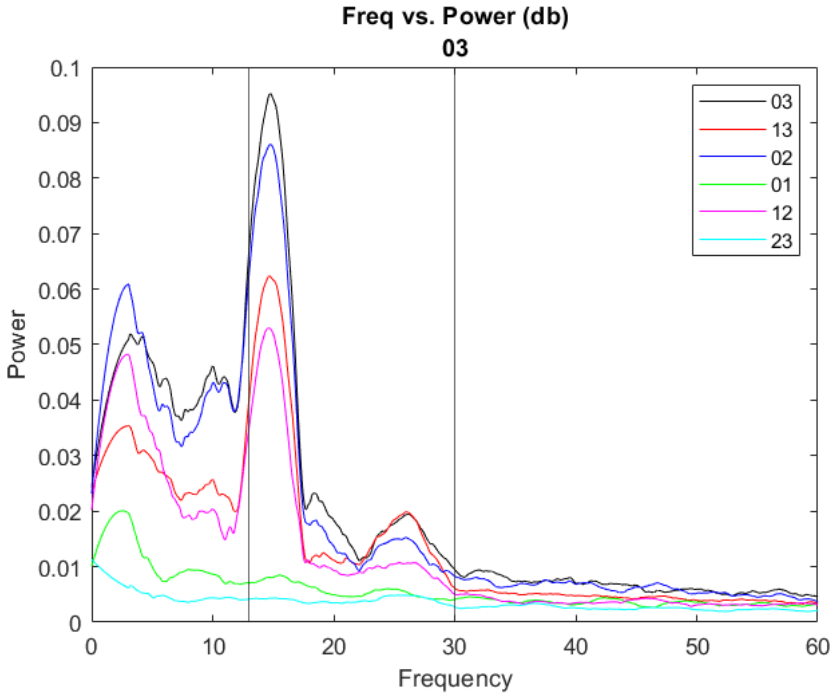
```
%% run to get full graph
%{} gets the thing from the actual cell array
%() gets cell ARRAY!

close

if doubleBattery == false
```

```
    %this should be true if the json returns both left and right data,
    %single battery
    for  i = 1:2
        subplot(2, 1, i)
        channels2 = channels(contains(channels,sides{i}));
        colors = 'krbgmc';
        maxValues = zeros(length(channels2), 1);
        for c=1:length(channels2)
            maxValues(c) = tempAvgPlot(startindex, leng, js, channels2{c},
colors(c));
            hold on;
        end

        [M, I] = max(maxValues);
        highestBeta = channels2{I};
        highestBetas{i} = highestBeta;

        %add all the plot info
        xline(13);
        xline(30);

        %formats the channels to be suitable for the legend
        legendChan = cell(6);
        oldchar = {'LEFT', 'RIGHT', '0', '1', '2', '3', '4', '5', '_', 'AND'};
        newchar = {''};
        oldnum = {'ZERO', 'ONE', 'TWO', 'THREE'};
        newnum = {'0', '1', '2', '3'};
        for b = 1:length(channels2)
            legendChan{b} = replace(channels2{b}, oldchar, newchar);
            legendChan{b} = replace(legendChan{b}, oldnum, newnum);
        end

        legend(legendChan{1}, legendChan{2}, legendChan{3}, legendChan{4},
legendChan{5}, legendChan{6})
        title(["Freq vs. Power (db)", sides{i}, legendChan{I}])
        xlim([0 60])
        xlabel("Frequency")
        ylabel("Power")

        disp(['The highest beta comes from contact pair ', legendChan{I}])


    end


 else
```

```
    %should be true if a double battery
    x = 0;
    %leng = numel({js.LfpMontageTimeDomain.Channel});
    %channels2 = channels(contains(channels,sides{i}));

    disp(["The following information is for the",
convertCharsToStrings(side)]);

    colors = 'krbgmc';
    maxValues = zeros(length(channels), 1);
    for c=1:length(channels)
        maxValues(c) = tempAvgPlot(startindex, leng, js, channels{c},
colors(c));
        hold on;
    end

    [M, I] = max(maxValues);
    highestBeta = channels{I};

    %add all the plot info

    xline(13);
    xline(30);

    %these are the order presented in channels

    legendChan = cell(6);
    oldchar = {'LEFT', 'RIGHT', '0', '1', '2', '3', '4', '5', '_', 'AND'};
    newchar = {''};
    oldnum = {'ZERO', 'ONE', 'TWO', 'THREE'};
    newnum = {'0', '1', '2', '3'};
    for b = 1:length(channels)
        legendChan{b} = replace(channels{b}, oldchar, newchar);
        legendChan{b} = replace(legendChan{b}, oldnum, newnum);
    end


    legend(legendChan{1}, legendChan{2}, legendChan{3}, legendChan{4},
legendChan{5}, legendChan{6})
    title(["Freq vs. Power (db)", legendChan{I}])
    xlim([0 60])
    xlabel("Frequency")
    ylabel("Power")

    disp(['The highest beta comes from contact pair ', legendChan{I}])
```

```
end
```

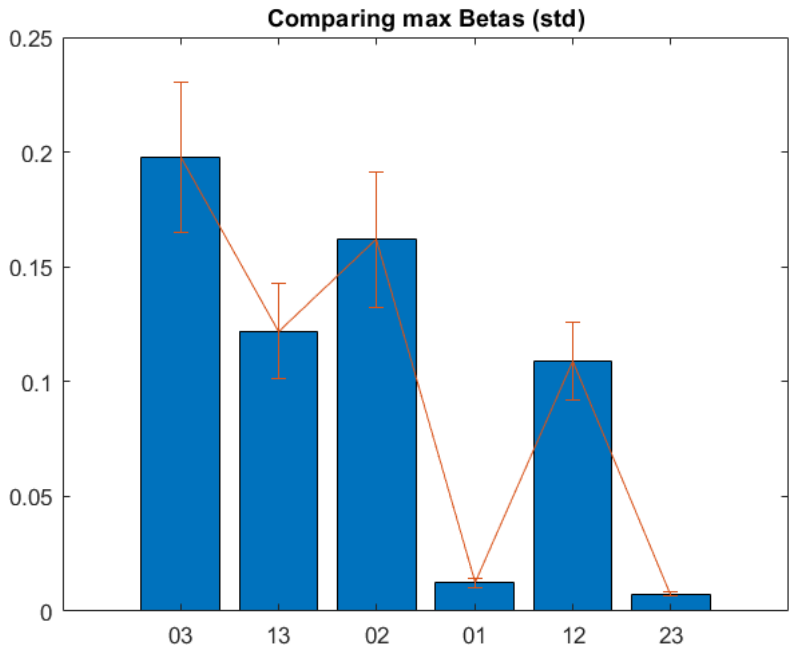
```
"The following information is for the"    "LEFT"
```




```
The highest beta comes from contact pair 03
```

Part 3: Compare beta values with stds across contacts

```
if doubleBattery == false
    for i = 1:2
        disp(['Max Betas ', sides{i}])
        compareBetaBarChart(startindex, leng, js, channelSides{i});
        figure;
    end
else
    compareBetaBarChart(startindex, leng, js, channels);
end
```



Part 4: Show table of data across each run

```
if doubleBattery == false
    for i = 1:2
        disp(['Table of means and stds per channel ', sides{i}])
        maxMeanBetaAllRuns(startindex, js, leng, channelSides{i});
        figure;
    end
else
    maxMeanBetaAllRuns(startindex, js, leng, channels);
    figure;
end
```

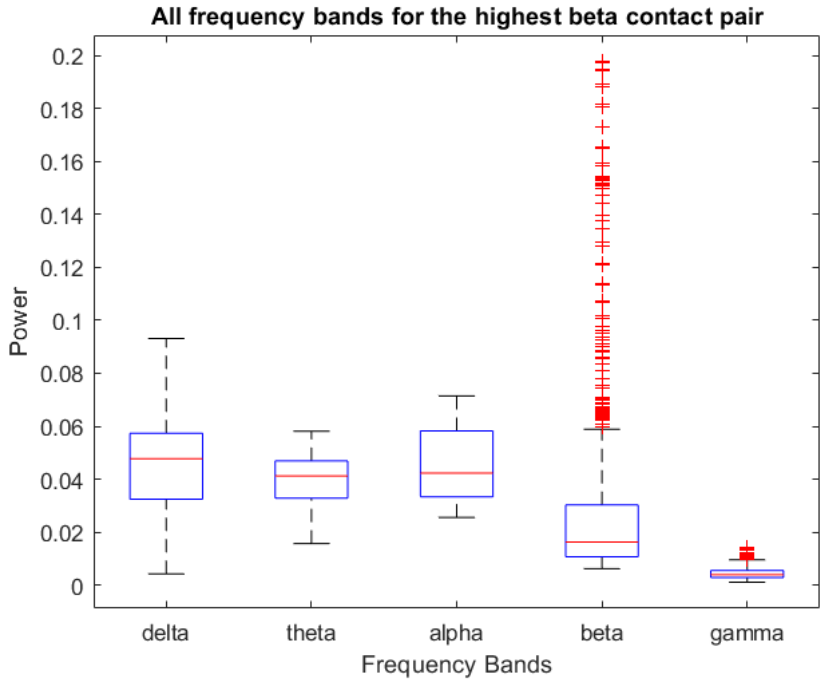
ans = 6×7 table

	Var1	Mean R1	Mean R2	Mean R3	STD R1	STD R2	STD R3
1	'ZERO_THREE_LEFT_0'	0.0314	0.0371	0.0269	0.0376	0.0520	0.0291
2	'ZERO_TWO_LEFT_2'	0.0297	0.0304	0.0222	0.0375	0.0422	0.0255
3	'ONE_AND_TWO_LEFT_1'	0.0226	0.0148	0.0158	0.0278	0.0133	0.0184
4	'ONE_THREE_LEFT_1'	0.0218	0.0266	0.0195	0.0264	0.0334	0.0174
5	'ZERO_AND_ONE_LEFT_0'	0.0064	0.0056	0.0056	0.0036	0.0022	0.0026
6	'TWO_AND_THREE_LEFT_2'	0.0040	0.0040	0.0041	0.0017	0.0018	0.0018

Part 5: Show all frequency data for contact pair that provides highest beta

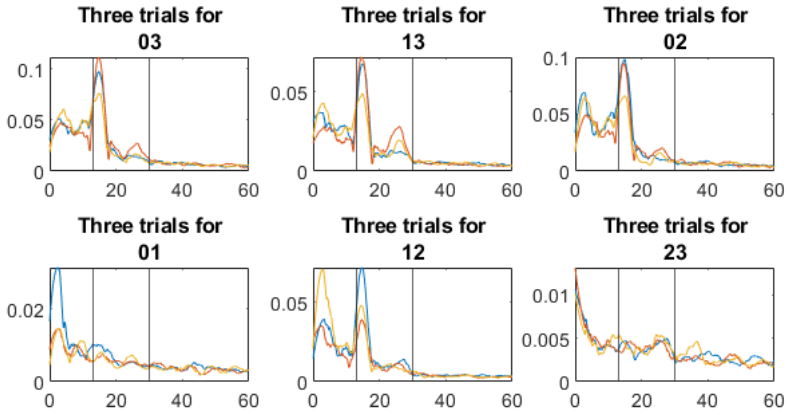
```
figure;
```

```
if doubleBattery == false
    for i = 1:2
        disp(['All frequency data for ', legendChan{I}, sides{i}])
        allFreqBarChart(startindex, leng, js, highestBetas{i});
        figure;
    end
else
    allFreqBarChart(startindex, leng, js, highestBeta);
    figure;
end
```



Part 6: Show variability between trials for each pair

```
%graphEachTrials(startindex, js, leng, channels);
figure;
if doubleBattery == false
    for i = 1:2
        disp(['Trial variability for the ', sides{i}])
        graphEachTrials(startindex, js, leng, channelSides{i});
        figure;
    end
else
   graphEachTrials(startindex, js, leng, channels);
   figure;
end
```



Perceive Medtronic-CU Anschutz Collaboration

RIGHT

Part 1: Import all necessary values and create global variables.

```
%% loads in json file
%loads in file
cd('C:\Users\sydne\Documents\github\perceive\patientData\Patient1_0524')
jsonFiles = 'Report_Json_Session_Report_20210524T101039.json';
js = jsondecode(fileread(jsonFiles));
%sets it back to the path where all the other functions are
cd('C:\Users\sydne\Documents\github\perceive')

%declares "global" variables AND determines if .json comes from
%single or double battery B)
channels = unique({js.LfpMontageTimeDomain.Channel}, 'stable');
leng = numel({js.LfpMontageTimeDomain.Channel});
startindex = 7; %MUST BE MANUALLY SET IN CLINIC
```

```
sides = {'LEFT', 'RIGHT'};
if any((contains(channels, sides{1}))) && any((contains(channels,
sides{2})))
    doubleBattery = false;
    %json has both left and right data
    channelsLeft = channels(contains(channels, sides{1}));
    channelsRight = channels(contains(channels, sides{2}));
    channelSides = {channelsLeft, channelsRight};
    highestBetas = {' ', ' '};
else
    doubleBattery = true;
    %added this to help with titling figures
    if contains(channels, sides{1})
        side = 'LEFT';
    else
        side = 'RIGHT';
    end
end
```

Part 2: Create freq vs pwr plot for all 6 contacts. Also reports the max betas.

```
%% run to get full graph
%{} gets the thing from the actual cell array
%() gets cell ARRAY!

close

if doubleBattery == false
    %this should be true if the json returns both left and right data,
    %single battery
```

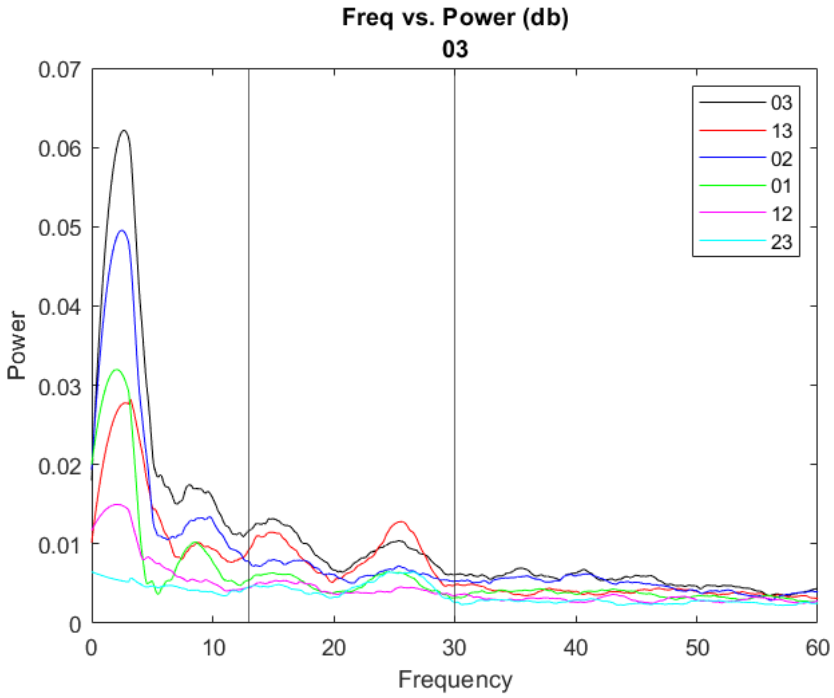
```
    for  i = 1:2
        subplot(2, 1, i)
        channels2 = channels(contains(channels,sides{i}));
        colors = 'krbgmc';
        maxValues = zeros(length(channels2), 1);
        for c=1:length(channels2)
            maxValues(c) = tempAvgPlot(startindex, leng, js, channels2{c},
colors(c));
            hold on;
        end

        [M, I] = max(maxValues);
        highestBeta = channels2{I};
        highestBetas{i} = highestBeta;

        %add all the plot info
        xline(13);
        xline(30);

        %formats the channels to be suitable for the legend
        legendChan = cell(6);
        oldchar = {'LEFT', 'RIGHT', '0', '1', '2', '3', '4', '5', '_',
'AND'};
        newchar = {''};
        oldnum = {'ZERO', 'ONE', 'TWO', 'THREE'};
        newnum = {'0', '1', '2', '3'};
        for b = 1:length(channels2)
            legendChan{b} = replace(channels2{b}, oldchar, newchar);
            legendChan{b} = replace(legendChan{b}, oldnum, newnum);
```

```
        end

        legend(legendChan{1}, legendChan{2}, legendChan{3}, legendChan{4},
legendChan{5}, legendChan{6})
        title(["Freq vs. Power (db)", sides{i}, legendChan{I}])
        xlim([0 60])
        xlabel("Frequency")
        ylabel("Power")

        disp(['The highest beta comes from contact pair ', legendChan{I}])


    end


 else
    %should be true if a double battery
    x = 0;
    %leng = numel({js.LfpMontageTimeDomain.Channel});
    %channels2 = channels(contains(channels,sides{i}));

    disp(["The following information is for the",
convertCharsToStrings(side)]);

    colors = 'krbgmc';
    maxValues = zeros(length(channels), 1);
    for c=1:length(channels)
        maxValues(c) = tempAvgPlot(startindex, leng, js, channels{c},
colors(c));
```

```
        hold on;
    end

    [M, I] = max(maxValues);
    highestBeta = channels{I};

    %add all the plot info

    xline(13);
    xline(30);

    %these are the order presented in channels

    legendChan = cell(6);
    oldchar = {'LEFT', 'RIGHT', '0', '1', '2', '3', '4', '5', '_', 'AND'};
    newchar = {''};
    oldnum = {'ZERO', 'ONE', 'TWO', 'THREE'};
    newnum = {'0', '1', '2', '3'};
    for b = 1:length(channels)
        legendChan{b} = replace(channels{b}, oldchar, newchar);
        legendChan{b} = replace(legendChan{b}, oldnum, newnum);
    end


    legend(legendChan{1}, legendChan{2}, legendChan{3}, legendChan{4},
legendChan{5}, legendChan{6})
    title(["Freq vs. Power (db)", legendChan{I}])
    xlim([0 60])
    xlabel("Frequency")
```

```
    ylabel("Power")

    disp(['The highest beta comes from contact pair ', legendChan{I}])
end
```

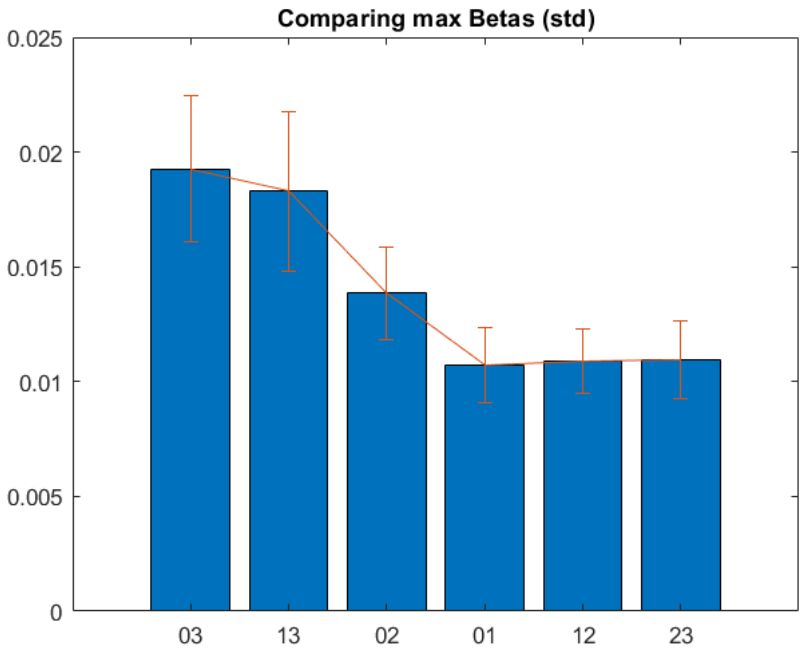
```
"The following information is for the"    "RIGHT"
```




```
The highest beta comes from contact pair 03
```

Part 3: Compare beta values with stds across contacts

```
if doubleBattery == false
    for i = 1:2
        disp(['Max Betas ', sides{i}])
        compareBetaBarChart(startindex, leng, js, channelSides{i});
        figure;
    end
else
    compareBetaBarChart(startindex, leng, js, channels);
end
```



Part 4: Show table of data across each run

```
if doubleBattery == false
    for i = 1:2
        disp(['Table of means and stds per channel ', sides{i}])
        maxMeanBetaAllRuns(startindex, js, leng, channelSides{i});
        figure;
    end
else
    maxMeanBetaAllRuns(startindex, js, leng, channels);
    figure;
end
```

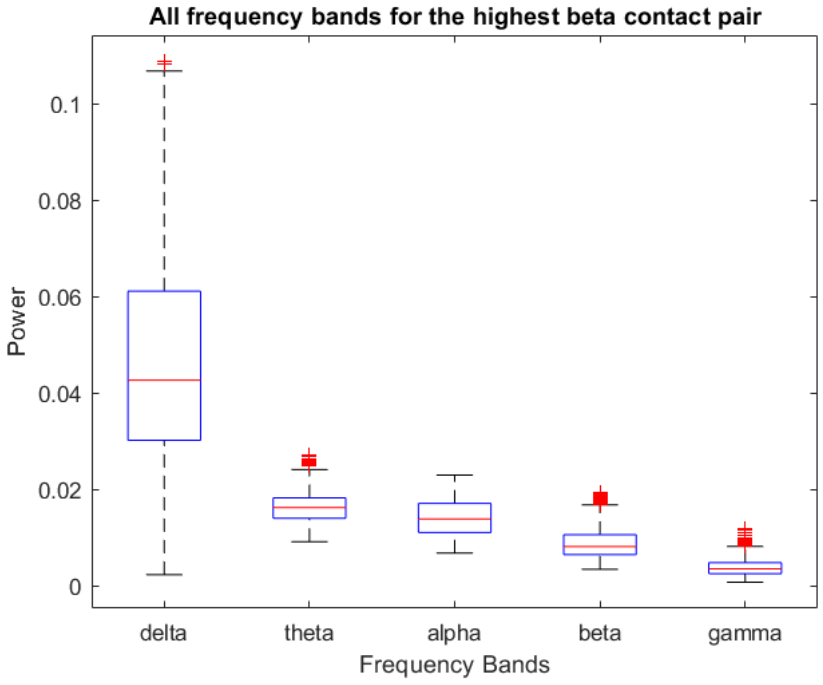
ans = 6×7 table

	Var1	Mean R1	Mean R2	Mean R3	STD R1	STD R2	STD R3
1	'ZERO_THREE_RIGHT_3'	0.0109	0.0086	0.0088	0.0056	0.0042	0.0045
2	'ONE_THREE_RIGHT_4'	0.0089	0.0083	0.0089	0.0053	0.0047	0.0048
3	'ZERO_TWO_RIGHT_5'	0.0070	0.0066	0.0065	0.0038	0.0029	0.0027
4	'ZERO_AND_ONE_RIGHT_3'	0.0051	0.0052	0.0049	0.0023	0.0029	0.0023
5	'ONE_AND_TWO_RIGHT_4'	0.0044	0.0041	0.0046	0.0022	0.0021	0.0024

	Var1	Mean R1	Mean R2	Mean R3	STD R1	STD R2	STD R3
6	'TWO_AND_THREE_RIGHT_5'	0.0041	0.0045	0.0055	0.0022	0.0023	0.0032

Part 5: Show all frequency data for contact pair that provides highest beta

```
figure;
if doubleBattery == false
    for i = 1:2
        disp(['All frequency data for ', legendChan{I}, sides{i}])
        allFreqBarChart(startindex, leng, js, highestBetas{i});
        figure;
    end
else
    allFreqBarChart(startindex, leng, js, highestBeta);
    figure;
end
```



Part 6: Show variability between trials for each pair

```
%graphEachTrials(startindex, js, leng, channels);
figure;
if doubleBattery == false
    for i = 1:2
        disp(['Trial variability for the ', sides{i}])
        graphEachTrials(startindex, js, leng, channelSides{i});
        figure;
    end
else
   graphEachTrials(startindex, js, leng, channels);
   figure;
end
```