**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\Patient3\_0630')

jsonFiles = 'Report\_Json\_Session\_Report\_20210630T155026.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 = 1; %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])

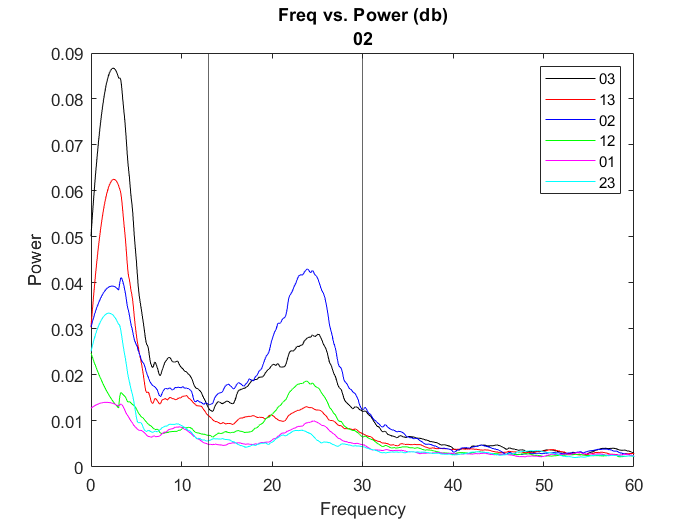
xlabel("Frequency")

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 02

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});

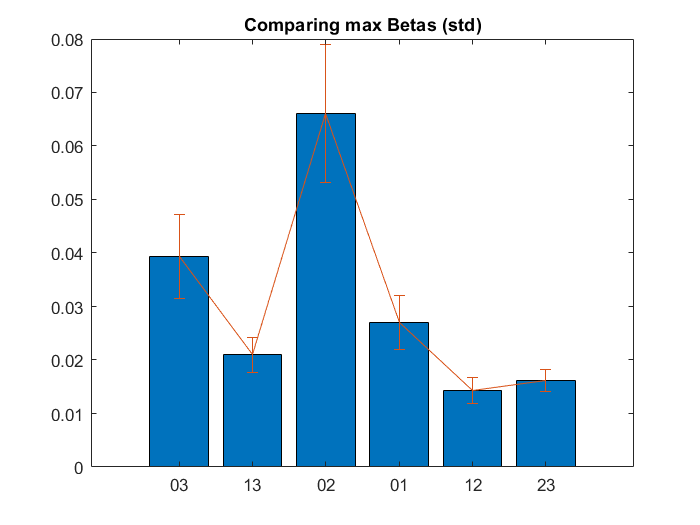
figure;

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\_AND\_TWO\_LEFT' | 0.0255 | 0.0289 | 0.0236 | 0.0163 | 0.0195 | 0.0146 |
| **2** | 'ZERO\_AND\_THREE\_LEFT' | 0.0201 | 0.0222 | 0.0167 | 0.0115 | 0.0133 | 0.0094 |
| **3** | 'ONE\_AND\_TWO\_LEFT' | 0.0111 | 0.0129 | 0.0099 | 0.0062 | 0.0080 | 0.0060 |
| **4** | 'ONE\_AND\_THREE\_LEFT' | 0.0109 | 0.0114 | 0.0089 | 0.0051 | 0.0055 | 0.0044 |
| **5** | 'ZERO\_AND\_ONE\_LEFT' | 0.0065 | 0.0067 | 0.0064 | 0.0036 | 0.0035 | 0.0034 |
| **6** | 'TWO\_AND\_THREE\_LEFT' | 0.0061 | 0.0054 | 0.0058 | 0.0032 | 0.0035 | 0.0029 |

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;

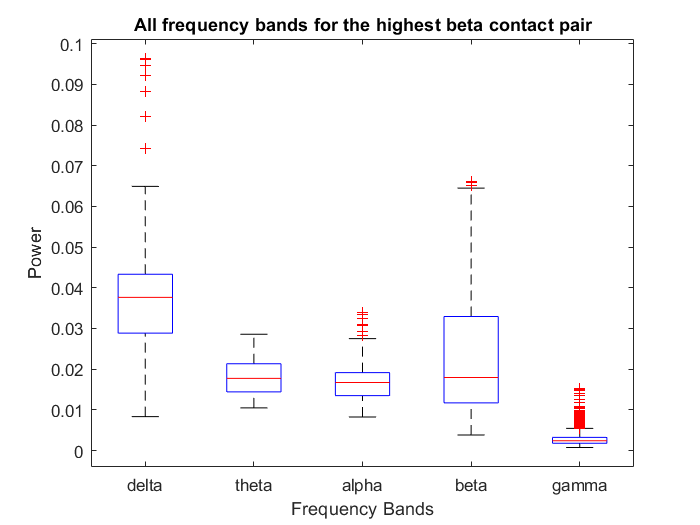
end

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;

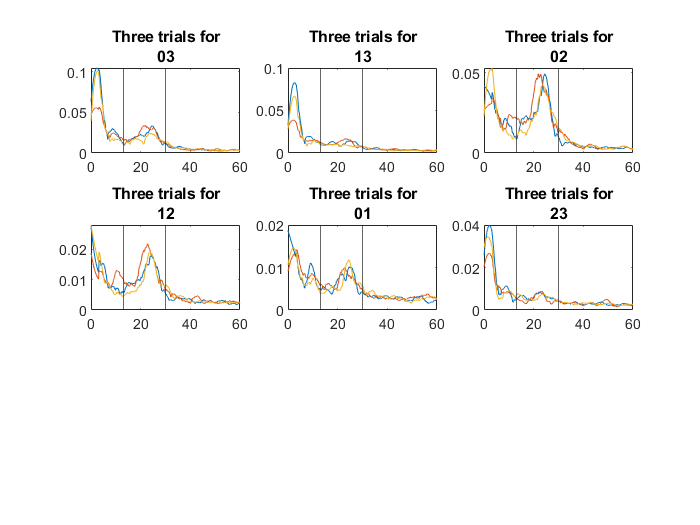
end

else

graphEachTrials(startindex, js, leng, channels);

figure;

end



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\Patient3\_0630')

jsonFiles = 'Report\_Json\_Session\_Report\_20210630T143145.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])

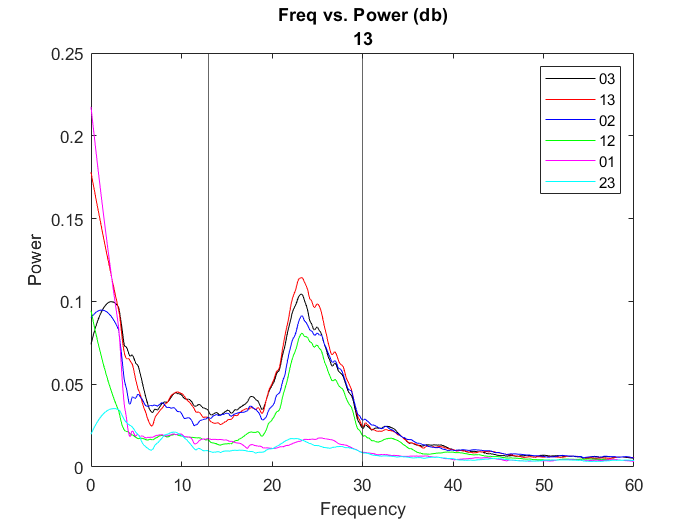
xlabel("Frequency")

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 13

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});

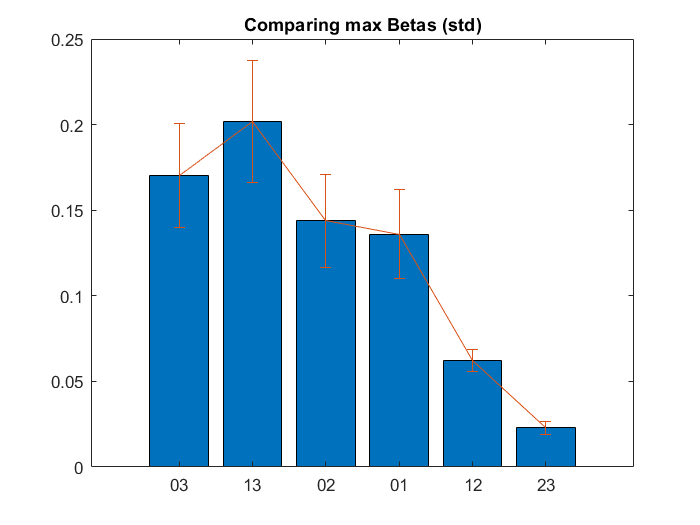
figure;

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** | **⋯** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | 'ZERO\_AND\_TWO\_RIGHT' | 0.0407 | 0.0507 | 0.0609 | 0.0250 | 0.0341 |  |
| **2** | 'ZERO\_AND\_THREE\_RIGHT' | 0.0391 | 0.0569 | 0.0717 | 0.0208 | 0.0388 |  |
| **3** | 'ONE\_AND\_THREE\_RIGHT' | 0.0378 | 0.0593 | 0.0782 | 0.0259 | 0.0454 |  |
| **4** | 'ONE\_AND\_TWO\_RIGHT' | 0.0286 | 0.0420 | 0.0468 | 0.0209 | 0.0349 |  |
| **5** | 'ZERO\_AND\_ONE\_RIGHT' | 0.0120 | 0.0132 | 0.0176 | 0.0082 | 0.0083 |  |
| **6** | 'TWO\_AND\_THREE\_RIGHT' | 0.0108 | 0.0103 | 0.0141 | 0.0051 | 0.0048 |  |

note: since the first Brainsense run on contact pair 13 does not produce the maximum beta, it is not listed as the pair with the highest beta in this table. However, both runs 2 and 3 produce the highest beta out of all the contact pairs.

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;

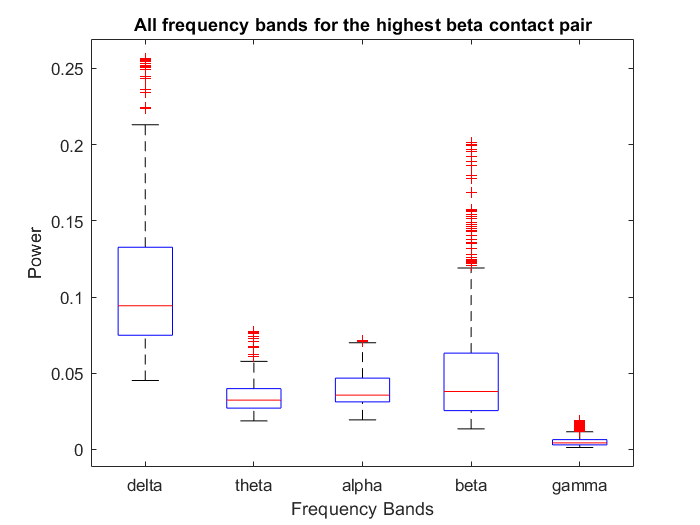
end

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

