On this document you can find a description of what each of the codes do.

On each folder there are main codes and all the files that you will need for them to run properly

**MATLAB 2020b** was used to run the MATLAB files and **Python 3.9.13** was used for the Python codes. **Spyder from Anaconda** was used to run the codes and display the plots.

**MATLAB Reaction Time & Accuracy**

**RT\_Acc\_KM.m**

The main objective of this code is to read the EMG files extracted during the handgrip task and to calculate both the reaction time and accuracy of each trial

The first part of the code has a list of the participants name, number of sessions and the MVC for each one of the participants.

Then, there are nested for loops in which we iterate between the subjects, the sessions (1,2,3), and the recordings (Baseline, during tACS, after 15 min & after 45 min). Up until this point, you won't know which type of stimulation each participant received because the files were named just with numbers and not the type of stimulation. If you want to know what stimulation they received on each session, go to the excel file named Participants.

On line 23, there is the path where the code should get the EMG files. You should change that to your own path where you saved them in your computer.

After that, the code loads all the info we need from each file.

The next few lines are to create a step signal that tells you when each event starts (when the participant was shown the bar and was supposed to start gripping)

After that, we downsample the data to 250 Hz. The next commented lines are supposed to plot a segment of the files to visualize the event step signal, the gripper, and both of the EMG signals. You can uncomment those lines if you want to visualize them but BE WARNED that there will be a lot of plots shown (15 subjects x 3 sessions x 4 recordings = 180 windows with plots) If you want to visualize only one, then just comment the lists at the very start and create new ones with just the subject and the session you want to visualize.

Then, accuracy is calculated and a variable called Error is created with 15 cells (each cell is a subject) then inside each cell there will be 3 more cells (corresponding to the sessions) and then inside of it will be 4 more cells (each recording) and then inside of each cell there will be all the 50 values of each trial for the accuracy. You have to save this variable on your computer for future use in a python script.

After that, there are some commented lines on which I tried to calculate the accuracy with different methods. In this case it was using the envelope of the EMG data.

Then, reaction time is calculated and saved in a variable called ReactionTime. It follows the same structure as the Error variable explained before.

The two variables Error and ReactionTime are the main outcomes that should be saved for future analyses.

**Plot\_RT\_Acc\_KM.m**

This code follows the same structure as the code before but in this case, the plot lines are not commented, so when you run it some windows will pop up corresponding to the subject selected on the variable SubjectName, from the session on the top list Sessions. If you want to view a different section of the data you can change the variables on line 74 & 75 where you specify the start and the end of the plots.

**MATLAB MRBDs**

**PlotMRBDs.m**

This code is for plotting the MRBD values pre-tACS, after 15 min and after 45 min for each session and each subject.

The first few lines are to create variables that have the subjects names, the number of Sessions and the recordings and the names of the electrodes that we want to plot. After that, we start declaring some other variables that will be needed in the following lines.

Then, the code starts with the for loops for iterating first the subjects, and then the sessions. Inside the second loop, we load the data from each MRBD for each recording (baseline, post-15 and post-45). Then, there’s another for loop, where it iterates between channels, the first if after the for loop is to check if the plot will be the data from a group level, then it checks if the channel is actually in the data, then it subplots each channel. If the data is subject level, it does the same procedure, subplotting each channel.

After ending all for loops, the final part of the code is for plotting the Young (from Xuanteng’s data) vs. Old data and it runs the same procedure, subplotting each channel.

**Python Reaction Time & Accuracy**

**RT\_Acc.py**

On this code, we plot the reaction time and accuracy that we calculated previously on the MATLAB code. Both on a subject level and group level.

First, we import all the libraries we are going to need. Then, we create a dictionary where we have the order in which each subject received the stimulation, 70 Hz, 20 Hz tACS or Sham stimulation. After that, we import the mat files previosly created in the MATLAB file, (Error and ReactionTime) and we convert them into data frames. Then we start iterating in for loops, for each subject, each session and each recording. In this first part we only work with the Error files. Inside the three for loops we first extract the outliers that we estimate as the values that are above 3 standard deviations of the mean. On the console, we print which outliers were removed, so you can keep track of all the removed values. After that we convert the data frames to jumpy arrays and then perform the t-tests of each comparison. It also prints the result for each t-test performed. After that, it creates boxplots on a subject level for each subject, for each session and all the recordings with significance bars if it detected a significant p-value in the t-tests.

After that, it creates new variables containing all the values from the 20 Hz, another for 70 Hz and another one for Sham condition. So, it creates different plots on a group level for each condition. It follows the same procedure for plotting boxplots and analyzing p-values.

The rest of the code follows the same procedure but for reaction time.

**RT\_Acc\_ANOVA.py**

On this code, we start importing the libraries and creating the same dictionary as before.

The code basically follows the same procedure as the previous one, importing the mat files, creating data frames and extracting outliers. At the end of the for loops, we just use the function AnovaRM which basically performs ANOVA using Time and Stimulation as factors, the output is in the console.

**Python Beta Power**

**BetaPower.py**

This code is for plotting the beta power of each channel, each subject, and each session. It follows the same logic as the previous code that was used for plotting the reaction time and accuracy. Th only main difference is that it also has a for loop that is used for iterating on each electrode. It also removes the outliers and it also plots on a group level after plotting on a subject level. At the very end of the code, it also creates a cat-plot so we can see on the same plot the behavior across time of beta power in all stimulations on a group-level.

**BetaPowerANOVA.py**

This code follows the same logic as the code before, but instead of plotting in also uses the function AnovaRM to conduct ANOVA tests on each electrode on a group level with factors Stimulation and Time. It displays the results on the console section.

**Python MRBD**

**MRBDs.py**

This code is basically the same as BetaPower.py but instead of beta power is the MRBD values. It plots on a subject and group level for each electrode and at the end it also plots a cat plot for all the stimulation conditions across time on a group level. It also prints the outliers and the results from the t-tests.

**MRBD\_Anova.py**

This code runs ANOVA tests on MRBD values the same way as before, using AnovaRM and it displays the results for each electrode on the console section.

**MRBD\_Correlation.py**

This code is used for creating Pearson’s correlation for each variable that we have. It imports the MRBD values the same way as the code before. It correlates with age, and also with motor performance (Reaction Time and Accuracy). It also creates correlation with the values from the behavioral motor results (PPT and BBT). It plots the correlation for all this measures.