HOW TO RUN
PREP CODE

Tutorial

What you'll need:

- The .csv files downloaded from Google Drive

Download from GitHub repository:

  - Matlab Script (OptCodeERPs.m).
  - Event files (ex: pitch8020.csv). Download the 'Events' folder, contains files for all events by condition.
  - Channel locations file (8_channels.txt).

Important: Add EEGLAB to the path in MATLAB and open it at least once before running the code.

File Preparation

Step 1 - Rename your files

After downloading the .csv files, change the names to this format so the code can find and match the files.

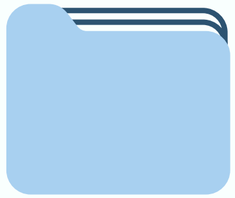
s1_pitch.csv

Setup

Step 2 - Adjust the Setup

2.1 – Set your working directoy:

- The code uses a variable called base_path to know where you're working.
- Inside that folder, manually create the following subfolders before running the code:



```
base_path/
│
├── Csv/          ← your renamed .csv EEG recordings
├── Mat/          ← intermediate .mat files
├── Raw/          ← .set files after importing channels & events
├── Prepro/       ← preprocessed .set files
│   └── ICA/      ← ICA-weighted .set files
```

Setup

Step 2 - Adjust the Setup

2.2 – Adjust the code setup:

- Set the correct **condition** you're working on.
- In **"num_sig"** set it to the your subject number, since you're only working with 1 at the moment.
- Make sure to also have your events folder and channel locations file in the right locations.

```
%% Setup
condition = 'pitch'; % CAMBIAR SEGUN VARIACION (pitch, duration, intensity, position)

base_path = 'C:\\Users\\pmb_0\\OneDrive\\NeuroTechs\\Registros EEG\\Registros T0';
csv_folder = fullfile(base_path, 'Csv');
mat_folder = fullfile(base_path, 'Mat');
input_filepath = fullfile(base_path, 'Raw');
output_filepath = fullfile(base_path, 'Prepro');
output_filepathICA = fullfile(output_filepath, 'ICA');
event_file = fullfile('C:\\Users\\pmb_0\\OneDrive\\NeuroTechs\\Registros EEG\\Events', [condition '8020.csv']);
chanlocs_file = 'C:\\Users\\pmb_0\\OneDrive\\NeuroTechs\\Registros EEG\\8_channels.txt';

num_sig = 3; % Num de sujetos
ica = cell(num_sig, 1);
```

Important

- The script was made to run multiple participants in one go with loops in most sections like:
  - *for i = 1:num_sig*
- But since in this case we're working on just one participant, you would have to change those loops. So, if your participant is number 3, change it to:
  - *for i = 3:num_sig*, and set *num_sig = 3* at the beginning.

CSV to SET

Step 3 - Convert EEG csv to EEGLAB format

```
%% .CSV to .SET conversion

for i = 1:num_sig
    subj_tag = sprintf('s%d_%s', i, condition);

    csv_filename = fullfile(csv_folder, [subj_tag '.csv']);
    mat_filename = fullfile(mat_folder, [subj_tag '.mat']);

    % Convert .csv to .mat
    dd = readtable(csv_filename);
    dd = rows2vars(dd);
    dd = dd(3:10, 1:50001);
    dd(:,1) = [];
    dd = table2array(dd);
    save(mat_filename, "dd", "-mat");

    % Load into EEGLAB, channels and events import
    EEG.etc.eeglabvers = '2023.1'; % EEGLAB version tracking
    EEG = pop_importdata('dataformat', 'matlab', 'nbchan', 0, 'data', mat_filename, ...
        'setname', subj_tag, 'srate', 250, 'pnts', 0, 'xmin', 0, 'chanlocs', chanlocs_file); % Import channels
    EEG = pop_importevent(EEG, 'event', event_file, 'fields', {'latency','type'}, 'skipline', 1, 'timeunit', 1); % Import events
    EEG = pop_saveset(EEG, 'filename', [subj_tag '.set'], 'filepath', input_filepath);
end
```

What happens:

- Reads your .csv file.
- Converts it to .mat.
- Loads it into EEGLAB.
- Imports events and channels.
- Saves as a .set file in the ‘Raw’ folder.

Preprocessing

Step 4 - Preprocess the EEG signal

What it does:

- Removes baseline.
- Applies high pass filter (1 Hz) to one dataset and obtains ICA weights.
- Filters (1-30 Hz) the original dataset and imports ICA weights from the high pass filtered dataset.
- Saves and prints ICA Averages.

```
%% Preprocessing

for i = 1:num_sig
    subj_tag = sprintf('s%d_%s', i, condition);

    % Load dataset for high-pass filter
    EEG = pop_loadset('filename', [subj_tag '.set'], 'filepath', input_filepath);

    % Remove baseline
    EEG = pop_rmbase(EEG, [], []);
    EEG_baseline_removed = EEG;

    % ICA weights on high-pass filtered dataset
    EEG = pop_eegfiltnew(EEG, 'locutoff', 1);
    EEG.setname = [subj_tag '_preproICA'];
    EEG = pop_runica(EEG, 'icatype', 'runica', 'extended', 1, 'interrupt', 'on');
```

```
ICA Component Averages:
    {'Signal S1 - Brain: 87.17%, Eyes: 1.40%, Muscles: 1.13%, Heart: 0.84%, Line Noise: 0.58%, Channel Noise: 1.44%, Others: 7.44%' }
    {'Signal S2 - Brain: 88.86%, Eyes: 0.12%, Muscles: 0.28%, Heart: 0.36%, Line Noise: 0.23%, Channel Noise: 0.07%, Others: 10.07%'}
    {'Signal S3 - Brain: 91.23%, Eyes: 0.05%, Muscles: 0.04%, Heart: 1.38%, Line Noise: 0.07%, Channel Noise: 0.05%, Others: 7.17%' }
```

Epoching

Step 5 - Cut the Signal around events

```
%% Epoching

all_epochs(num_sig) = struct('freq', [], 'infreq1', [], 'infreq2', [], 'infreq3', [], 'infreq4', []);

for i = 1:num_sig
    subj_tag = sprintf('s%d_%s_prepro', i, condition);
    filename = [subj_tag '.set'];

    % Load dataset
    s_eeg = pop_loadset('filename', filename, 'filepath', output_filepath);
    suj = ['Subject ', num2str(i)];

    % Frequent
    dd_freq = pop_epoch(s_eeg, {'Frequent'}, [-0.2 0.6]);
    dd_freq = pop_rmbase(dd_freq, [-200 0], []);

    % Infrequent 1
    dd_infreq1 = pop_epoch(s_eeg, {'1-Infrequent'}, [-0.2 0.6]);
    dd_infreq1 = pop_rmbase(dd_infreq1, [-200 0], []);

    % Infrequent 2
    dd_infreq2 = pop_epoch(s_eeg, {'2-Infrequent'}, [-0.2 0.6]);
    dd_infreq2 = pop_rmbase(dd_infreq2, [-200 0], []);

    % Infrequent 3
    dd_infreq3 = pop_epoch(s_eeg, {'3-Infrequent'}, [-0.2 0.6]);
    dd_infreq3 = pop_rmbase(dd_infreq3, [-200 0], []);

    % Infrequent 4
    dd_infreq4 = pop_epoch(s_eeg, {'4-Infrequent'}, [-0.2 0.6]);
    dd_infreq4 = pop_rmbase(dd_infreq4, [-200 0], []);
```

What it does:

- Finds the 5 types of events: Frequent, 1-Infrequent, 2-Infrequent, 3-Infrequent and 4-Infrequent.
- Extracts time windows (-200 to 600 ms)
- Removes baseline from each epoch

ERP calculation & Plotting

Step 6 - Calculate and view ERPs

The last section computes and plots mean ERPS. You get:

- Top: frequent stimuli ERP.
- Bottom: infrequent stimuli ERPs.

The colors are selected from the color palette defined in the setup section, with each assigned number.