Georgia Institute of Technology, Atlanta (GA), United States

School of Electrical and Computer Engineering

Essential Motor Cortex Signal Processing: an ERP and functional connectivity MATLAB toolbox - User Guide Version 1.0

Esmaeil Seraj[†,*]

Karthiga Mahalingam[†]

[*] Corresponding author

[†]Correspondences shall be forwarded to: *electronic mail:* {eseraj3, kmahalingam}@gatech.edu

July 8, 2019

Abstract

The purpose of this document is to help individuals use the "Essential Motor Cortex Signal Processing MATLAB Toolbox". The toolbox implements various methods for three major aspects of investigating human motor cortex from Neuroscience view point: (1) ERP estimation and quantification, (2) Cortical Functional Connectivity analysis and (3) EMG quantification. The toolbox – which is distributed under the terms of the GNU GENERAL PUBLIC LICENSE as a set of MATLAB® routines – can be downloaded directly at the address:

`http://oset.ir/category.php?dir=Tools`

.

or from the public repository on GitHub, at address below:

`https://github.com/EsiSeraj/ERP_Connectivity_EMG_Analysis`

.

The purpose of this toolbox is threefold:

1. Extract the event-related-potential (ERP) from preprocessed cerebral signals (i.e. EEG, MEG, etc.), identify and then quantify the event-related synchronization/desynchronization (ERS/ERD) events. Both time-course dynamics and time-frequency (TF) analyzes are included.
2. Measure, quantify and demonstrate the cortical functional connectivity (CFC) across scalp electrodes. These set of functions can also be applied to various types of cerebral signals (i.e. electric and magnetic).
3. Quantify electromyogram (EMG) recorded from active muscles during performing motor tasks.

Key-words: Event-Related Potential, ERP, Event-Related Synchronization, ERS, Event-Related Desynchronization, ERD, ERP Time Dynamics, Time-Frequency Analysis, Cortical Functional Connectivity, CFC, Electroencephalogram, EEG, Electromyogram, EMG, EMG Quantification, MATLAB Function, Free Toolbox, User Guide, Manual

Contents

1 Introduction

This document is meant to help individuals use the "Essential Motor Cortex Signal Processing MATLAB Toolbox". Using the most popular reference articles in literature, the toolbox implements various methods for three major aspects of neuro-physiological investigation of human motor cortex: (1) ERP estimation and quantification (e.g. based on [1, 2, 3, 4]), (2) Cortical Functional Connectivity analysis (e.g. based on [5, 6, 7, 8, 9]) and (3) EMG quantification (e.g. based on [10, 11]).

The purpose of this toolbox is threefold:

1. Extract the event-related potential (ERP) from preprocessed cerebral signals (i.e. EEG, MEG, etc.), identify and then quantify the event-related synchronization/desynchronization (ERS/ERD) events. Both time-course dynamics and time-frequency (TF) analysis are included.
2. Measure, quantify and demonstrate the cortical functional connectivity (CFC) across scalp electrodes. These set of functions can also be applied to various types of cerebral signals (i.e. electric and magnetic).
3. Quantify electromyogram (EMG) recorded from active muscles during performing motor tasks.

A primary goal of this toolset is to ease understanding the routines and help individuals alter our codes according to their study and/or implement their own techniques. For this purpose, herein, we first present a detailed tutorial (e.g. Section 2.2.) on signal preconditioning and interpretable-implementation of introduced methods, from a practical view-point

1.1 License - No Warranty

1.1.1 The Open-Source Electrophysiological Toolbox (OSET)

Open Source Electrophysiological Toolbox (OSET) is a collection of electrophysiological data and open source codes for biosignal generation, modeling, processing, and filtering. OSET, version 3.1, 2014 Released under the GNU GENERAL PUBLIC LICENSE [12]. Copyright© 2012.

As a progressive general-purpose open-source toolset, OSET is one of the main sources to access the functions and codes of current *motor cortex signal processing* toolbox and also its documentation and dependencies.

1.2 Citation

Within the limits of the GNU GENERAL PUBLIC LICENSE, you can use the toolbox as you please; however, if you use the toolbox in a work of your own that you wish to publish, you need to make sure to cite this user manual and the original studies properly, as shown below. This way you will contribute to helping other scholars find these items.

- Esmaeil Seraj and Karthiga Mahalingam "Essential Motor Cortex Signal Processing: an ERP and functional connectivity MATLAB toolbox - User Guide Version 1.0," arXiv Preprint, June 2019 [Online].

1.3 Download and Utilization

The latest version of the toolbox can be downloaded directly from OSET at the address:

`http://oset.ir/category.php?dir=Tools`

.

or from the public repository on GitHub, at:

`https://github.com/EsiSeraj/ERP_Connectivity_EMG_Analysis`

.

The functions and m-files can be downloaded separately as you need or all together in a compressed file named `ERP_Connectivity_EMG_Analysis_Toolbox`. Once you have downloaded and uncompressed the toolbox, 35 files represented in following Tables shall be appeared in your chosen directory. Table (1) represents all m-files (i.e. core functions, internal computational functions and demo test files) included in this toolset with their short descriptions and references. Additionally, a copy of both GNU general public license, this user manual and ten recorded sample data are also included (see Section 2.2.1 for details). Moreover, you might add the path of the directory in which you stored the toolbox to your MATLAB in order to easily use and apply the functions for your own dataset.

1.4 Getting Help

If you have added the toolbox directory to the MATLAB® path you can simply type:

`<doc function_name>` or `<help function_name>`

in command window to get online help for the function you are using. Furthermore, you can also contact any of the authors[1] directly to ask any related questions or discuss possible difficulties or errors you might encounter. Please feel free to contact in either case.

[1]E. Seraj, K. Mahalingam: {`eseraj3, kmahalingam`}`@gatech.edu`

Table 1: Essential Motor Cortex Signal Analysis m-files

row	name (function)	description	reference
1	`test_ERPanalysis_main.m`	demo for testing the ERP analysis functions	Original
2	`test_connectivity_main.m`	demo for testing the functional connectivity functions	Original
3	`test_EMGanalysis_main.m`	demo for testing the EMG quantification functions	Original
4	`trigger_avg_erp.m`	trigger-averaged ERP time-course estimation function	Original
5	`trigger_avg_TF_erp.m`	trigger averaged ERP time/frequency representation	Original
6	`erp_quantification.m`	quantifying the area of ERD/ERS events	Original
7	`TCPLV.m`	time-course PLV estimation function for a pair of electrodes	Original
8	`PWPLV.m`	pair-wise PLV estimation function	Original
9	`PWCoherence.m`	pair-wise coherence estimation function	Original
10	`PLV_PhaseSeq.m`	PLV quantification function	OSET [13]
11	`emg_quantification.m`	quantifying the EMG signals	Original
12	`phase_est.m`	TFP phase estimation function	OSET [13]
13	`trigger_synch.m`	synchronizing signals according to trigger time	Original
14	`bdf2mat_main.m`	reading EEG/EMG signal from bdf files, preprocessing and conditioning	Original
15	`emg_onset.m`	EMG onset estimation function	Original
16	`drift_reject.m`	drift cancellation from biological recordings	OSET [12]
17	`BaseLine2.m`	estimating the baseline of a signal	OSET [12]
18	`sig_trend.m`	estimating the trend of a signal	OSET [12]
19	`BPFilter5.m`	forward-backward zero-phase CIC filtering function	OSET [12]
20	`task_separator.m`	task based data separation to avoid memory overuse	Original
21	`eeg_read_bdf.m`	Converting “*.bdf” to “*.mat”	Gleb Tcheslavski [14]

.

2 User Guide

2.1 Overview

This document is meant to help individuals use the "Essential Motor Cortex Signal Processing MATLAB Toolbox" which implements various methods for three major aspects of investigating human motor cortex from Neuroscience view point: (1) ERP estimation and quantification (e.g. based on [1, 2, 3, 4]), (2) Cortical Functional Connectivity analysis (e.g. based on [5, 6, 7, 8, 9]) and (3) EMG quantification (e.g. based on [10, 11]). Measurements and methodologies are all derived based on the procedures suggested by the most popular reference articles in that category, as aforementioned.

The current version of toolbox covers six well-known and widely used approaches within Neuroscience community for analyzing motor cortex potentials and EMG signals, as follows:

- **Preconditioning:** Preprocessing the signals, including baseline drift removal, artifact rejection, movement onset detection (i.e. for non-cued movements), trigger synchronization (i.e. essential step for ERP extraction through non–synchronized trials of EEG data) and etc.
- **Time-course ERP Dynamics:** Estimation and quantification of ERP (i.e. ERS and ERD events).
- **Time-frequency ERP Analysis:** Estimation and representation of time-frequency (TF) maps through various widely-accepted TF analysis methods such as Short-time Fourier Transform (STFT), Continuous Wavelet Transform (CWT), Narrow-band Channelization (NBCH) and etc.
- **Time-course Cortical Functional Connectivity:** Estimation and quantification of local/large -scale functional connectivity between arbitrary pairs of electrodes through Phase Locking Value (PLV) [7, 9] and Magnitude Squared Coherence (MSC) [6].
- **Pair-wise Cortical Functional Connectivity:** Estimation and quantification of local/large -scale functional connectivity maps across scalp electrodes through PLV and MSC.
- **EMG Quantification:** Estimation and quantification of electromyogram (EMG) records of active muscles during performing motor tasks.

2.2 Fundamentals

2.2.1 Data: Setup, Equipments and Format

ActiveTwo is a very well-known commonly used high-resolution biosignal acquisition system that comes with advantageous capabilities such as active electrodes [15]. Active electrodes are those that does not require extensive skin (scalp) preparation due to the presence of pre-amplifiers in them. The pin electrodes must be connected to the scalp on one end and to the acquisition system on the other. The acquisition system is then connected to the PC to which it relays electrode data serially. Electrode gel has to be used on the scalp with electrodes to improve conductivity and to reduce noise. Care should be taken to reduce noise as much as possible by properly grounding all electrical appliances connected with the electrode system, making sure there are minimum head movements while recording etc.

Before starting data collection, all electrodes need to be checked for their proper functionality. This can be performed through the single bucket test [16]. Electrodes can be placed according to the 10-20 system [17], which is a very common approach for most studies. EEG cap has to be adjusted based on head circumference, nasion-inion, and ear-to-ear distance. Reference electrode areas are normally chosen to be the average between two ears [15] and thus CMS/DRL flat type electrodes should be placed behind earlobes. Finally, EMG electrodes should be placed directly on the active muscle the activity of which is to be recorded.

The EEG signals recorded by ActiveTwo are stored in *.bdf format files. We provided a very useful MATLAB m-file by Gleb Tcheslavski [14] that converts *.bdf format files to *.mat MATLAB data files.

2.2.2 Preconditioning

In ERP and functional connectivity analysis problems, preconditioning mostly refers to preprocessing the signals. This includes baseline drift removal, artifact rejection, movement onset detection from EMG signals, trigger synchronization (i.e. as described above) and etc.

Quick note on artifact rejection: The concept of noise in this problem could be regarded to two different parts of recorded signals. First group are the common sources of noise observed in EEG recordings such as muscle activity, eye movement, electrocardiographic activity, instrumentation and equipment related artifacts and interference, slow drifts and amplifier saturation and etc. [2]. Moreover, background EEG activity is considered as the other source of noise [1]. In this case, ERP is considered as signal and the background EEG activity as noise [2, 1].

Artifact rejection in later case is normally easy, since as widely accepted, background ongoing EEG is considered as uncorrelated to signal (ERP) and thus, can be rejected by an ensemble averaging over trials [1, 2, 3]. The best way of dealing with the first group of noises however, is to not have any in first place. The easiest sources of noise to deal with are AC power lines, lighting and electronic equipment such as computers, displays and TVs, wireless routers, notebooks and also mobile phones. The basic step here is to simply remove any unnecessary sources of electro-magnetic (EM) noise from the recording room and, if possible, replace equipment using alternate current with direct current [18, 19]. Further, a significant number of noise sources can be rejected through different steps of low-pass, high-pass and sometimes notch filters.

Drift cancellation: Despite having various sources of additional noise on EEG, one of the most important ones to deal with, in this application, is the baseline drift noise (a.k.a base line wander). The drift noise normally happens due to sweating, drifts of cap and/or electrodes and similar reasons, which could normally lead to amplifier saturation [2, 18] and thus incorrect and unreliable measurements. The effect of drifts on ERP estimation application is shown in Fig. 1 which is directly employed from [2]. As shown, although the effect of drift removal from 182 trials is relatively small, still is considerable for analyzing the relative decreases and increases in ERP power (ERD/ERS) quantification.

As suggested in [2], a number of issues are to be taken into consideration, such as detecting negative drifts as well as positive ones, using similar specs for the drift rejection method across trials and also choosing a proper-length (long enough) temporal window for drift calculation (preferably between 1 or two seconds) [2]. Nevertheless, the filtering specs in EEG artifact removal are highly dependent on the data and recording conditions and thus, should be chosen empirically [2, 20]. The strongest

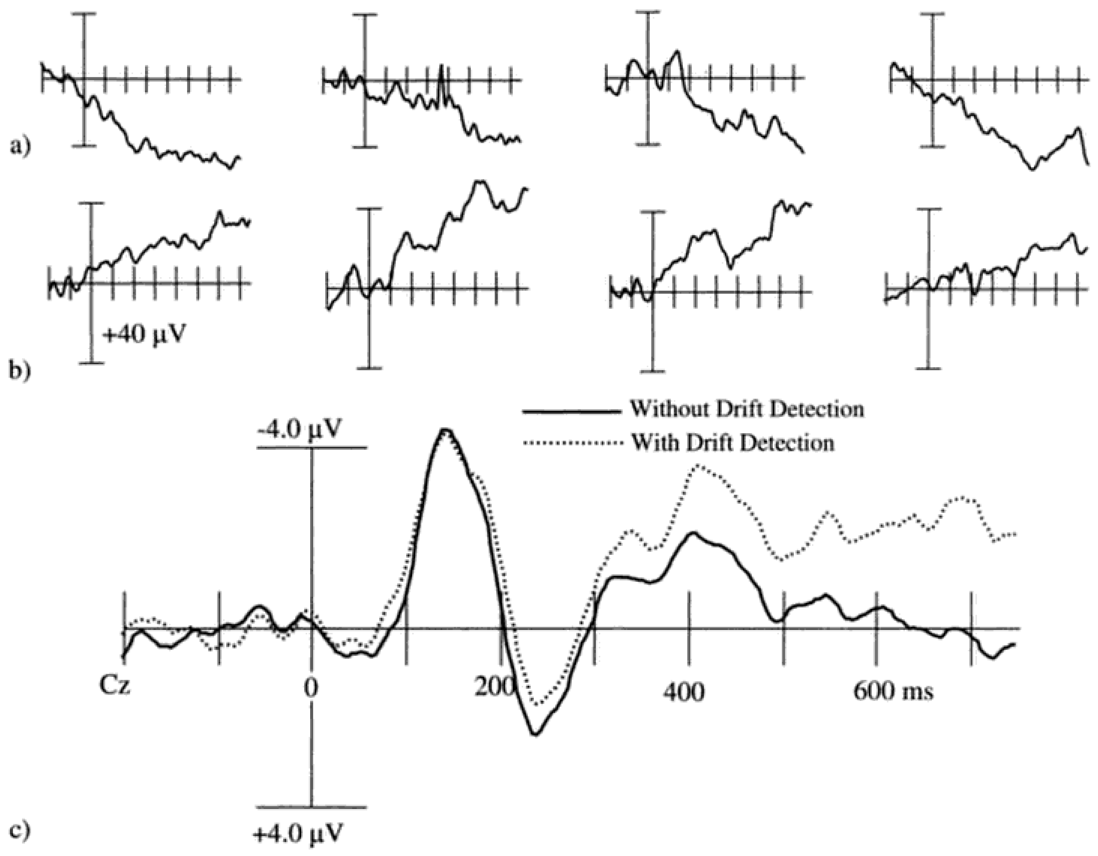

Figure 1: The effect of drift (base line wander) on ERP analysis. (a) and (b) examples of trials with drifts. (c) estimated ERP in each case. This figure is adopted from [2].

recommendation throughout the literature is to evaluate the filter before becoming committed to [2, 20]. An important consideration to be taken into account here is that the filtering procedure itself, depending on the characteristics and nature of the utilized filter and filtering method, can also significantly affect the data in the non-filtered frequency ranges and thus can affect the onset detection and/or amplitude of estimated ERP. Therefore, to prevent these problems, it is highly recommended that filtering should be limited to what is necessary and unavoidable [18].

In this toolbox, the function `drift_reject.m` is provided for this purpose which uses two stages of median or moving average temporal filtering to extract the base-line wander and reject it. To operate this function you can simply call function below with proper input parameters:

```
sig = drift_reject(raw_sig, L1, L2, approach)
```

where the inputs and outputs are described later in Section 3. Fig. 2 is an elaboration of how how this function works.

2.2.3 Time-course ERP Dynamics

For this part, the conventional approach, as proposed in [1], is implemented. Accordingly, the well-known five-step method presented below is employed for ERP extraction from EEG signals:

- **Step #1** – Band-pass frequency filtering the EEG data in each trial
- **Step #2** – Squaring the samples of filtered trials to find the power

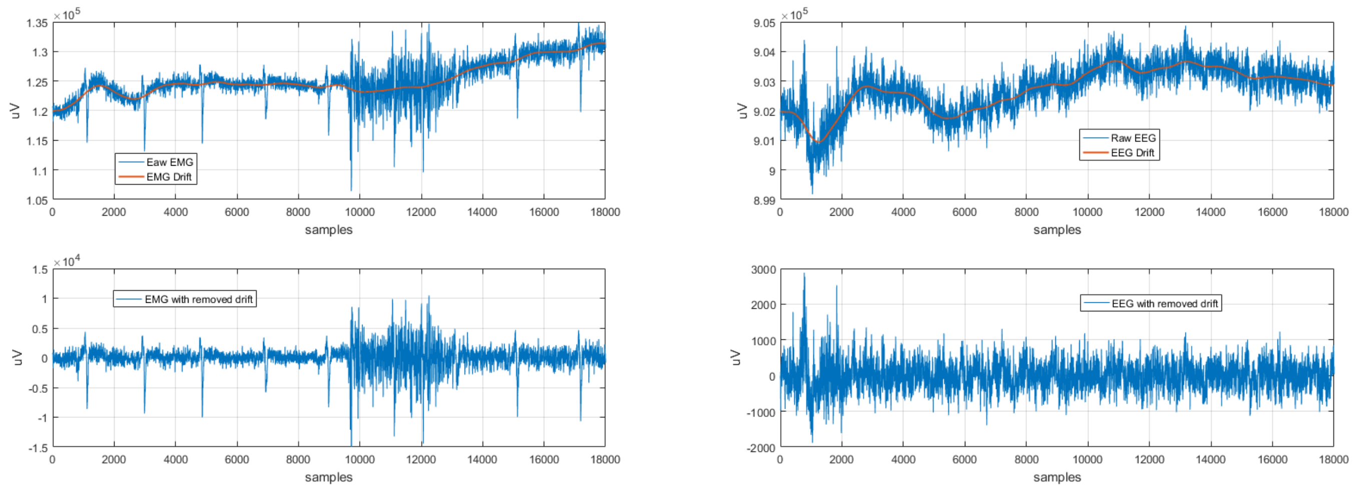

Figure 2: Two sample EMG (left) and EEG (right) from provided sample dataset before and after removing the drift artifact as described in [2, 18, 20]

- **Step #3** – Ensemble averaging over synchronized trials according to trigger time
- **Step #4** – Calculating the trend of output sequence in order to obtain ERP signal
- **Step #5** – Quantifying the estimated ERP signal according to a reference period

The estimated signal in Step #4 is the desired ERP signal. Each step is elaborated below.

Step #1: For this step, zero-phase bandpass frequency filtering using Cascaded Integrator-Comb (CIC) filter is implemented [21]. The filter performs forward-backward filtering successively. It has absolutely zero-phase for *even* filter order values and a phase-lag equal to a single stage CIC filter for *odd* filter order values. Therefore, for this application we recommend an even value to be chosen as the filter order to prevent any subsequent phase distortion.

An important issue here is to select the frequency band of interest for showing the time course of ERP [1]. This concern is very crucial due to the fact that different brain states can modulate different frequency bands of EEG rhythms. Additionally, even for the same action, one might observe intra-trial variability on dominant frequencies [1]. Accordingly, estimating the time-course of ERP signal related to the task for further analysis requires additional cautions to be taken into account.

Due to the subjective process of choosing a proper frequency band for ERP analysis, herein, the functions are designed to accept frequency ranges in a way that users can customize their frequency range of interest and experiment various bands of frequency easily. To read further in these regards, refer to [1, 2, 3].

Steps #2, #3 and #4: Steps 2 through 4 are closely related (e.g. even could be considered as one step) where after squaring the samples of the filtered signal, two successive *averaging* steps are performed across trials and then time [1]. Ensemble averaging is the most important one among these steps since EEG signals are required to be averaged over synchronized data trials according to their respective trigger time. Based on this, first the onset of movement needs to be estimated (i.e. can be performed either through hardware settings (e.g. cue based recordings) or using our provided `emg_onset.m` function) as the trigger time. Then, the EEG trials have to be sorted and aligned with respect to the trigger time and eventually averaged across data trials. Afterwards, a simple temporal averaging is performed on obtained signal to extract the ERP sequence.

Interested individuals can refer to either the Reference Manual (i.e. Section3) or [1, 2, 3] for details of each averaging step and implemented signal trend estimation function.

Steps #5: For this step, normally a 1-sec long reference period before the movement onset is chosen. Afterwards, the relative decrease intervals in power after movement onset are called ERD events and vice versa, the relative increase intervals in power are named ERS [22, 23].

According to discussions presented in [22, 23, 1], reference period is chosen some seconds before the task onset, completely based on application of interest and data and thus is empirically set. Reference value is calculated as the average of power samples within reference period. An axis according to the reference value is set to 100% and the corresponding crosses of ERP curve are named respectively, time step "0", start of ERD1/ERS1, end of ERD1/ERS1 and so on. Afterwards, standard deviation of the reference period is calculated and the corresponding confidence intervals are defined (mean plus/minus 2 or 3 times STD). Finally, the area enclosed by ERP curve and "**mean - confidence interval**" is calculated and normalized by the length of that segment as ERD area and vice versa; the area enclosed by ERP curve and "**mean + confidence interval**" is calculated and normalized by the length as ERS area.

In this toolbox, the function `trigger_avg_erp.m` is provided for this purpose which utilizes `trigger_synch.m` function to synchronize cerebral signals according to their trigger time and `BPFilter5.m` function to perform the aforementioned CIC filtering. To operate this function you can use the demo m-file `test_ERPanalysis_main.m` or simply call function below with proper input parameters:

```
[erp, synch_eeg, synch_emg, trigger_time_sec, time_vec] = trigger_avg_erp(eeg, emg,
                        fs, emg_onset_sampl, freq_band, duration)
```

where the inputs and outputs are described later in Section 3. Fig. 3 is an elaboration of how how this function works.

Moreover, function `erp_quantification.m` is provided to quantify the calculated ERP signal as the output of `trigger_avg_erp.m` according to the exact procedure described above. To operate this function you can simply call function below with proper input parameters:

```
[erd_area, ers_area, quant_erp] = erp_quantification(erp, fs, trigger_time_sec,
                                  ref_per, cof_intv)
```

where the inputs and outputs are described later in Section 3. Fig. 4 is an elaboration of how how this function works.

2.2.4 Time-frequency ERP Analysis

Time-frequency representations (TFRs) can monitor ERPs in a wide range of frequency components alongside with their temporal variations. Using TFRs, not only the risk of missing important information is lower, but also the interpretations provided can be more comprehensive and generic. Herein, as a part of ERP analysis package, a MATLAB function is dedicated to this purpose in which three different approaches for time/frequency analysis, namely Short-Time Fourier Transform (STFT), Continues Wavelet Transform (CWT) and Narrow-Band Channelization method (NBCH) as introduced in [1], are implemented.

Theoretical concepts of each TFR method is out of scope of the current document, however, interested users can refer to [24] for a detailed and comprehensive discussion on time/frequency analysis concepts and its techniques. Nevertheless, as a short description, in STFT, the TF map of each trial is

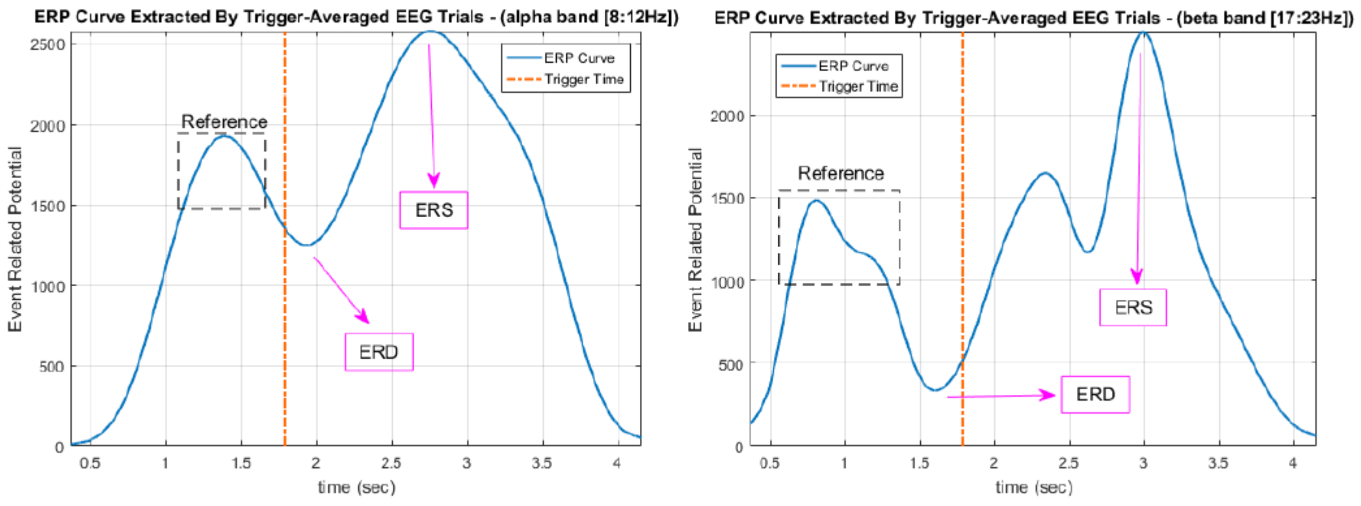

Figure 3: Time courses of trigger averaged ERP curves, related to alpha and beta bands.

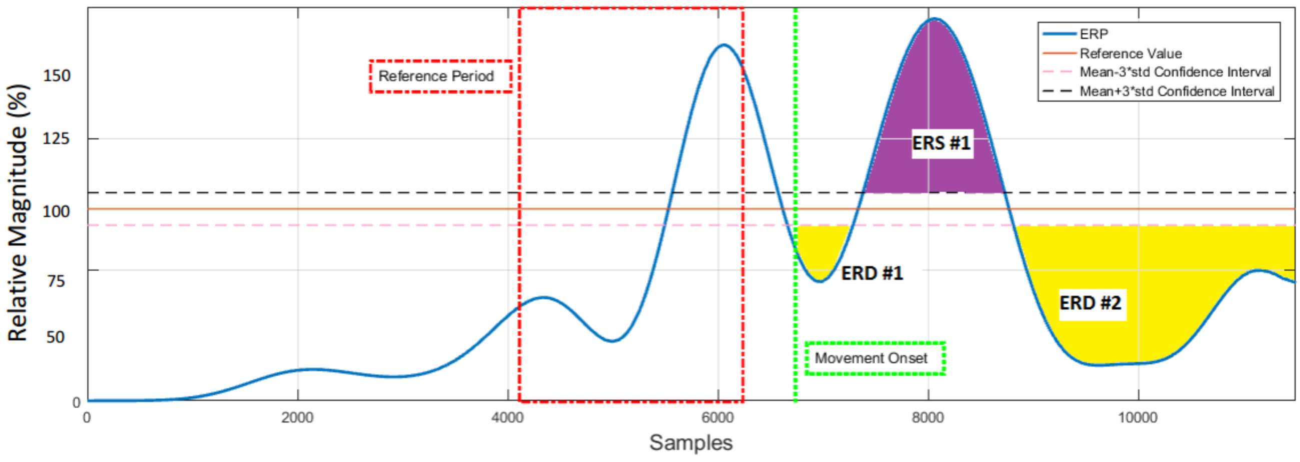

Figure 4: Procedure for quantifying the area of ERD/ERS Events. Note that the confidence interval lines are plotted with exaggeration for better representation and are not in scale.

calculated separately through Fourier Transform and averaged over all. Similarly, for CWT the scalogram displaying the squared and over-all-trials-averaged wavelet coefficients are used. Eventually, for NBCH, ERP maps are generated according to ERP behaviors within very narrow frequency bands. The ERP maps are matrices, the rows of which correspond to frequency-specific ERP estimations.

The implemented function enables variable frequency range. The procedures utilized for all three TFR methods here are in accordance with [1]. The function `trigger_avg_TF_erp.m` is provided for this purpose which utilizes `trigger_ synch.m` function to synchronize cerebral signals according to their trigger time and `BPFilter5.m` function to perform the aforementioned CIC filtering for NBCH method. Moreover, for STFT and CWT methods, MATLAB's inner `spectrogram` and `cwt` functions are used. To operate this function you can use the demo m-file `test_ERPanalysis_main.m` or simply call function below with proper input parameters:

```
[erp_tf, synch_eeg, synch_emg, trigger_time_sec, freq_vec, time_vec] =
 trigger_avg_TF_erp(eeg, emg, fs, emg_onset_sampl, duration, method)
```

where the inputs and outputs are described later in Section 3. Fig. 5 is an elaboration of how how this function works. It is worth noting that the CWT produces similar results. Note that several benchmark studies such as [25], [26], [27] and [28] are research show-cases of the real-world applications how these TF analysis data can be put to use. For instance, as investigated and shown by [29] and also [27] and [30] inspecting the irregularities of the TF patterns through entropy quantities can be very informative regarding the underlaying motor-task [25], depth of sleep and Neurons activity rate [29, 27], levels of Alzheimer's disease [28] and even the cognitive skills and their respective performances in subjects [31, 26, 13].

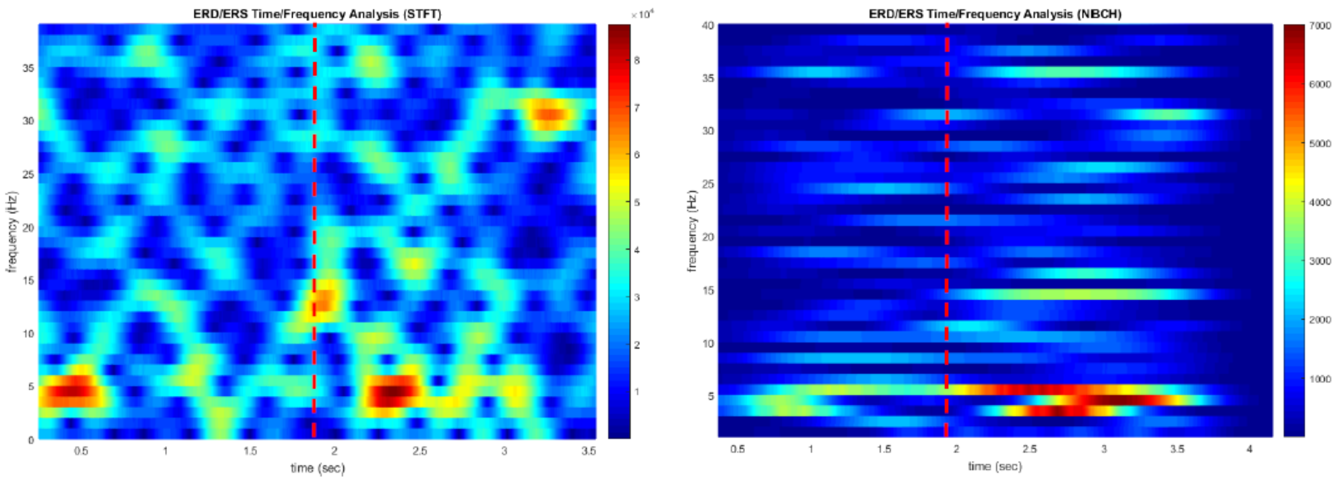

Figure 5: TF maps estimated through STFT and NBCH methods. Note that despite the visible differences, both methods show almost same interpretations (note the relative magnitudes), with respect to the trigger time (red dashed line).

2.2.5 Time-course Cortical Functional Connectivity

Time-course cortical functional connectivity is usually used to investigate the local-scale brain connectivity across different electrodes within same brain regions (i.e. primary motor cortex). To this end, Phase Locking Value (PLV) as the common quantity for this purpose [7, 9, 32] are implemented, measuring both temporal dynamics of local-scale and heat-maps of large-scale functional connectivity. The details of each method is described below.

Phase-locking Value (PLV): PLV is a measure for quantifying how constant the phase difference between two signals is. In order to calculate the PLV in frequency f for two signals (or channels) $x(t)$ and $y(t)$, the following steps are required [7, 9]:

- Using narrow-band filters centered at f, calculate the instantaneous frequency-specific phase values $\phi_x(t, f)$ and $\phi_y(t, f)$.
- Calculate the instantaneous phase-difference between $x(t)$ and $y(t)$ and quantify the local stability of this phase-difference over time as follows:

$$PLV(f) = \left| \frac{1}{T} \sum_{t=1}^{T} \exp\left(j[\phi_y(t, f) - \phi_x(t, f)]\right) \right| \tag{1}$$

  where T is the signal length and the summation is over all temporal samples of the instantaneous phases.

PLV varies between 0 and 1, corresponding to completely non-synchronized signals and complete synchronization, respectively [7, 9]. In simple wolds, considering the difference between firing rates of neurons as phase difference between their electrical potentials, PLV measures how ”*connected*” different neurons in far regions of brain are through quantifying the difference between their firing rates [7, 9, 32].

One very important consideration is the phase estimation approach that one needs to take into account. For phase estimation in first step, the Transfer Function Perturbation (TFP) method introduced in [33] and [34] has been adopted. TFP is a statistical Monte-Carlo based phase estimation approach which uses infinitesimal perturbations on filter and other signal estimation parameters in order to account for stochastic properties of EEG signals. According to the discussion presented in Section 2.2.2, using TFP is specifically required due to presence of background EEG noise (which in this case cannot be rejected as simple). It has been shown in [33] and [34] that without using TFP, estimated phase of EEG may contain variations and spikes which do not have any physiological origins within brain or are not due to brain activity and are merely side-effects of estimation method. For this purpose, cerebral signal phase estimation toolbox [13] has been used.

In this toolbox, function `TCPLV.m` is provided for this purpose which first utilizes `phase_est.m` function to estimate the instantaneous phase (IP) sequence of input signals through introduced TFP method and then uses `PLV_PhaseSeq.m` function to quantify the phase difference between calculated IPs. Function `TCPLV.m` measures time-course PLV for ***1sec time-steps*** either between two specified electrode channels or between one important channel and all other channels (resulting in a heat-map across scalp). As recent examples of using such framework and implementation (time-course PLV) we can mention [34, 27, 25]. To operate this function you can use the demo m-file `test_connectivity_main.m` or simply call function below with proper input parameters:

```
tcplv = TCPLV(eeg, fs, freq_rng, pairofint, emg_onset_sampl, duration, pertnum)
```

where the inputs and outputs are described later in Section 3. Figures 6, 7 and 8 are elaborations of how this function works in either of aforementioned cases. Read the figure captions for more details.

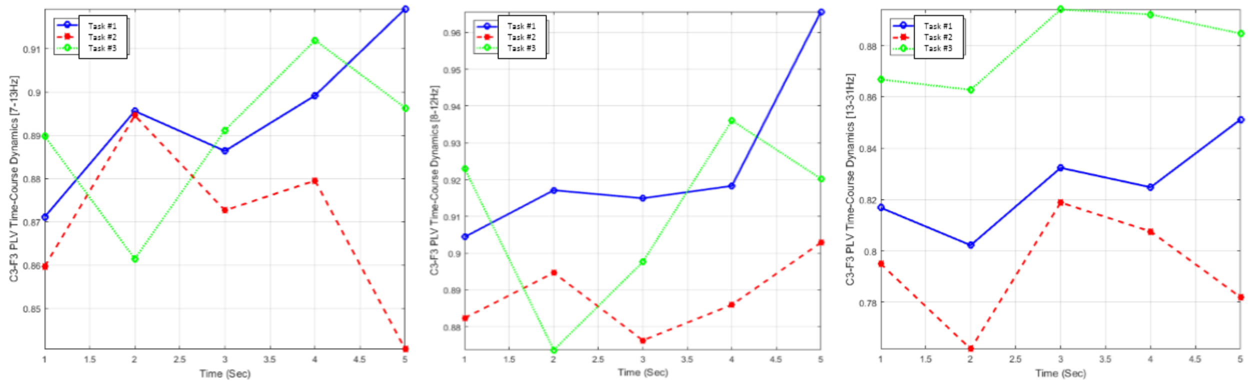

Figure 6: C3-F3 PLV time-course dynamics for three different motor tasks in a clinical experiment measured for 7-13Hz, 8-12Hz and 13-30Hz rhythms from left to right respectively, within 1 second intervals. Movement onset is around 3.5sec.

2.2.6 Pair-wise Cortical Functional Connectivity

Pair-wise cortical functional connectivity is usually used to investigate the large-scale brain connectivity across scalp (i.e. between different brain regions). To this end, the Magnitude Squared Coherence

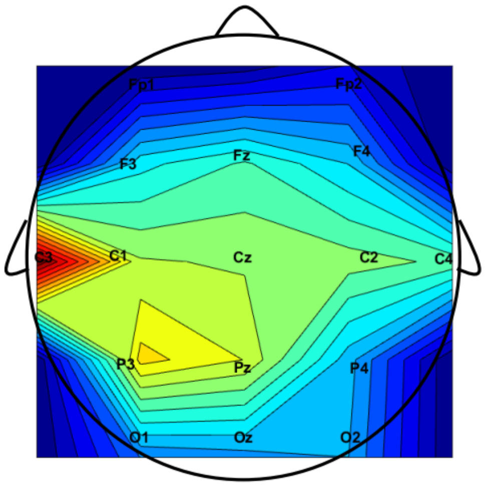

Figure 7: Electrode placements for time-course PLV maps between C3 and all other EEG channels.

(MSC) and Phase Locking Value (PLV) as the common quantities for this purpose [6, 35, 7, 9, 32] are implemented, measuring heat-maps of large-scale functional connectivity.

Magnitude Squared Coherence (MSC): The conventional method presented in [6] for measuring coherence is based on weighted windowing of the Fourier transform of signals [32]. Considering $x(t)$ and $y(t)$ as two channels of recorded EEG signals, MSC can be measured in frequency f as below:

$$|MSC(f)|^2 = \frac{|PSD_{xy}(f)|^2}{PSD_{xx}(f)PSD_{yy}(f)} = \frac{|\sum_{i=1}^{N} X_i(f)Y_i^*(f)|^2}{\sum_{i=1}^{N} |X_i(f)|^2 \sum_{i=1}^{N} |Y_i(f)|^2} \tag{2}$$

The MSC here, is calculated using a non-overlapping hamming window using FFT analysis for EEG, in a frequency range of interest for all trials, and then averaged across trials. The mean MSC then is analyzed within ***1 second temporal windows*** covering -3 to -2, -2 to -1, -1 to 0, 0 to 1 and 1 to 2 seconds, with respect to 0 set as trigger time. MSC measures are calculated for all possible electrode combinations, resulting in N×N MSC maps (for N-channel EEG recording), representing the functional connectivity across the whole scalp. Similar to PLV, MSC varies between 0 to 1 [6, 32, 36].

In this toolbox, function `PWCoherence.m` is provided for this purpose which first utilizes `BPFilter5.m` function to perform the aforementioned CIC filtering and then uses MATLAB's internal `mscohere.m` function to quantify the frequency-specific MSC values. As mentioned, function `PWCoherence.m` measures MSC maps for ***1sec time-steps*** between all possible pairs of electrode. As recent examples of using such framework and implementation (pair-wise MSC) we can mention [26, 27, 25, 34]. To operate this function you can use the demo m-file `test_connectivity_main.m` or simply call function below with proper input parameters:

```
pwcoher = PWCoherence(eeg, fs, freq_rng, emg_onset_sampl, duration, plot_flag)
```

where the inputs and outputs are described later in Section 3. Figures 9 are elaborations of how how this function works in either of aforementioned cases. Read the figure captions for more details.

Phase Locking Value (PLV) for Large-scale Functional Connectivity: Similar to MSC, PLV can be measured between all possible electrode pairs, resulting in PLV maps across entire scalp. We provided this option in this toolbox by implementing function `PWPLV.m` which first utilizes

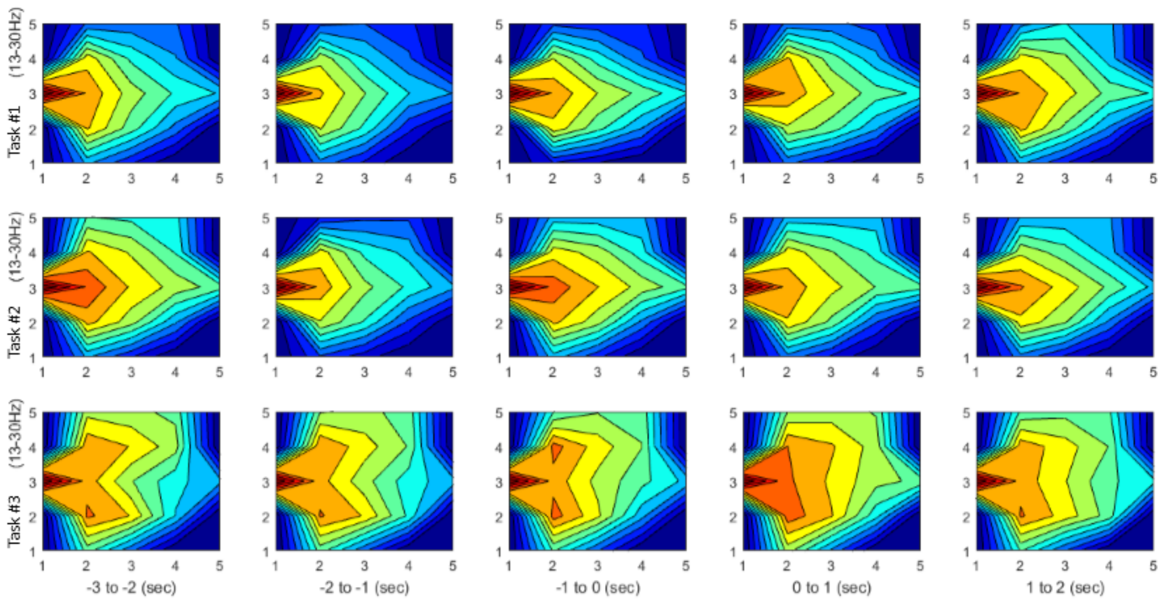

Figure 8: Normalized time-course PLV maps between C3 and all other electrodes for beta rhythms (13-30Hz) for 1 second time bins from -3 prior to 2 seconds after the movement onset.

`phase_est.m` function to estimate the instantaneous phase (IP) sequence of all channels of input data through introduced TFP method and then uses `PLV_PhaseSeq.m` function to quantify the phase difference between calculated IPs. Function `TCPLV.m` measures pair-wise PLVs for ***1sec time-steps*** between all possible electrode pairs resulting in N×N PLV maps (for N-channel EEG recording), representing the functional connectivity across entire scalp. As recent examples of using such framework and implementation (time-course PLV) we can mention [33, 34, 27, 25, 26]. To operate this function you can use the demo m-file `test_connectivity_main.m` or simply call function below with proper input parameters:

```
pwplv = PWPLV(eeg, fs, freq_rng, emg_onset_sampl, duration, pertnum, plot_flag)
```

where the inputs and outputs are described later in Section 3. This function also outputs figures similar to Figures 9.

2.2.7 EMG Quantification

The purpose of this section of toolbox is to estimate and quantify electromyogram (EMG) signals recorded from active muscles during performing motor tasks. The EMG quantification is a common practice within Neuroscience community in order to evaluate the work-load of brain (through EMG curve's immediate slope after movement onset) and also for the rehabilitation purposes (stronger muscles after brain stimulation or training). The basic steps for EMG quantification are as below [10]:

- Removing Electrocardiogram (ECG) artifacts from EMG signals of each trial
- Form the full-rectified signal from the remaining EMG signal of each trial
- Trigger-average the rectified signals across trials
- Extract the trend of the averaged signal as the quantified EMG

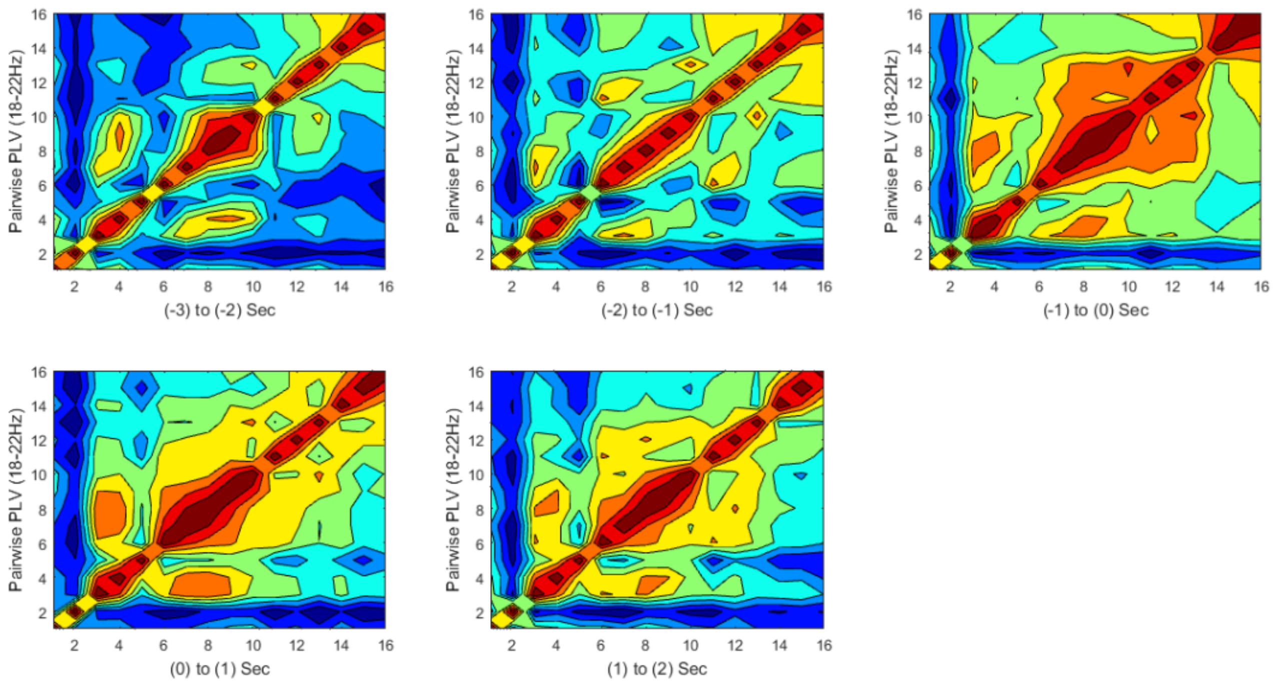

Figure 9: MSC maps within 1 second time steps for an upper-body limb movement task.

The first step of this process, i.e. removing the ECG artifact, can be processed through several different simple or more complex approaches [37]. High-pass frequency filters (HPF) [38], ECG subtraction through QRS-complex detection (FAS) [39], adaptive filtering (AF) approaches [40, 41], ICA based approaches [42, 43] and also combined AF-ICA based methods [44]. Here in this toolbox however, we promise to stick with the simple and fast methods and thus, implement a modified high-pass frequency filtering approach not only to gain from the simplicity but to also improve the performance accuracy of removing ECG artifact from EMG signals. Accordingly, our modified ECG removal approach includes low-pass filter combined with median filtering (**LPF+MF**). HPF is normally used only to remove the major ECG frequency components which are restricted by the (1-30Hz) [45]. The main drawback of this approach is due to the fact that HPF will remove the frequency components of EMG too [45]. Nevertheless, to overcome this limitation, we implemented and used a LPF+MF to first detect the ECG signal (instead of blindly removing its components) and then perform a temporal subtraction. To this end, first we track the major ECG frequency components (1-30Hz) with a fourth order (can be modified within function parameters) elliptic IIR low-pass filter with pass-band ripple and stop-band attenuations of 0.1dB and -50dB, respectively. As before, the filtering is processed in a forward-backward zero-phase manner to avoid any phase distortion due to non-linear phase response of IIR filters. Afterwards, a median temporal filter with ***small*** window-length is used to estimate and extract the ECG patterns more accurately [10, 45, 46, 47].

In this toolbox, function `emg_quantification.m` is provided for this purpose which first utilizes MATLAB's internal `ellip` function to generate an elliptic IIR filter and then uses `BaseLine2.m` function twice; first to track and remove the ECG components and then to quantify the EMG signal as described before. To operate this function you can use the demo m-file `test_EMGanalysis_main.m` or simply call function below with proper input parameters:

```
[emg_bl, synch_emg2, ecg_estimate2_bl, time_vec] = emg_quantification(emg_data, fs,
                              emg_onset_sampl, duration)
```

where the inputs and outputs are described later in Section 3. Fig 10 is an elaboration of how this function works. Performance of simple HPF is also shown for comparison and elaborating the reliability of the propose approach (i.e. LPF+MF). Maximum magnitude (muscle activation strength), activation slope (muscle activation speed) and immediate activation slope after movement onset (muscle-brain work load) are some potentially important information captured from quantified EMG through our provided function. See Fig 11 for graphical elaboration of these concepts.

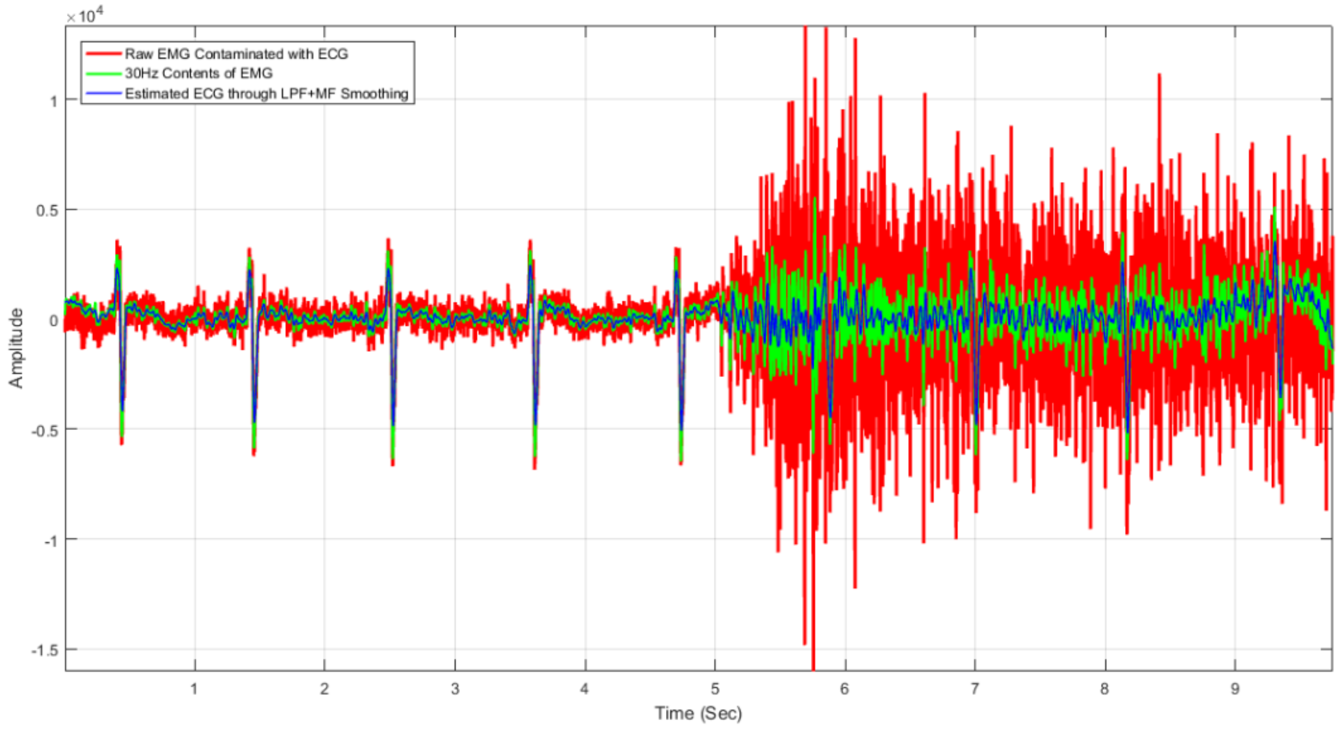

Figure 10: The aforementioned procedure suggested to extract the ECG patterns from EMG signal. Note the final blue curve as the extracted ECG signal. The blue curve is then subtracted from the red one to form the ECG-free EMG signal.

EMG Movement Onset Detection: As promised before in Section 2.2.3, here, we provide users with a simple but fast and reliable EMG onset estimation function. The onset detection in EMG signals has a broad literature in which various methods aimed to satisfy different purposes and applications are proposed and discussed. The simplest proposed approach for this purpose is the single thresholding method [48] which later on has been modified into double-threshold methods [49, 50]. Although there are several complex highly sensitive methods for onset detection, their utility really depends on users' application of interest and thus to reduce computational cost, here we designed and implemented a simple two-stage thresholding method which is highly reliable for ERP application. The proposed two-stage thresholding method is basically formed based on double-thresholding framework (which includes three steps) [49, 50]. The three steps are (1) signal conditioning (which is done in preprocessing phase), (2) detection of an event (performed by calculating a windowed standard deviation over time, named as STD vector) and (3) exact detection of the movement time (performed by using a threshold on STD vector's temporal trend). As reported previously in [49] and [50], moving temporal filtering and thresholding methods are very common and could be satisfactory based on application requirements. Additionally, according to discussions in [49] and [50], parameters and thresholds in such approaches need to be set empirically based on the data and application.

In this toolbox, function `emg_onset.m` is provided for this purpose which follows the exact procedure as described above. To operate this function you can use the demo m-file `test_EMGanalysis_main.m`

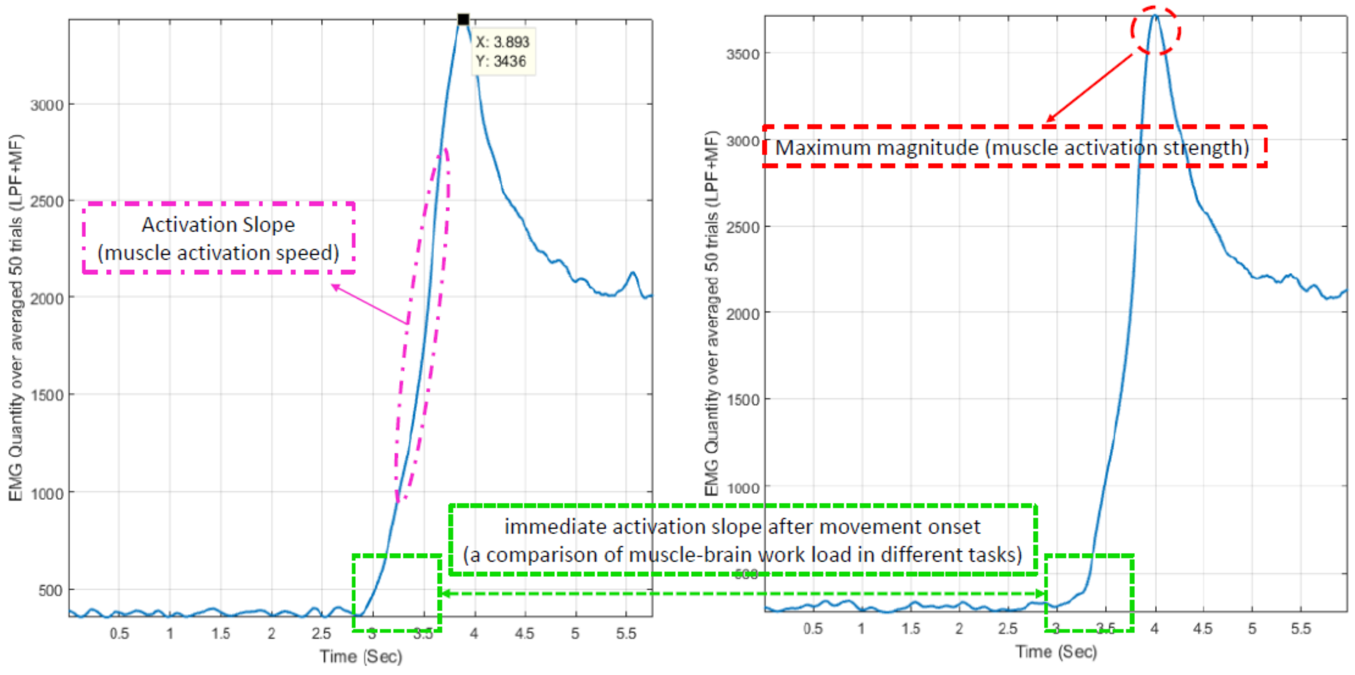

Figure 11: Quantified EMG curves over 50 trials of two different tasks. Note the represented important parameters derived from quantified EMG.

or simply call function below with proper input parameters:

```
[onset_sampl, onset_time] = emg_onset(emg, fs, W, th_coeff, Trl)
```

where the inputs and outputs are described later in Section 3. Fig 12 is an elaboration of how this function works. It worth noting that, although the implemented procedure is not as sensitive as complex new methods, it still satisfies our requirements for the application of ERP analysis [46, 47]. This is due to the fact that ERP is calculated by averaging over a large number of trials and thus here we are interested in **intra-trial** variations and information rather than **inter-trial** ones.

3 Reference Manual

3.1 bdf2mat_main.m

```
bdf2mat_main.m
```

Purpose: Reading EEG/EMG signal from bdf files, preprocessing and conditioning.

Synopsis (global mode):

```
[eeg_data, emg_data, fs] = bdf2mat_main(trl_num, elec_num, emg_ch, eeg_ch, filename)
```

Synopsis (local mode):

```
[eeg_data, emg_data, fs, emg_onset_sampl, emg_onset_time] = bdf2mat_main(trl_num,
        elec_num, emg_ch, eeg_ch, filename, drift_flag, onset_flag)
```

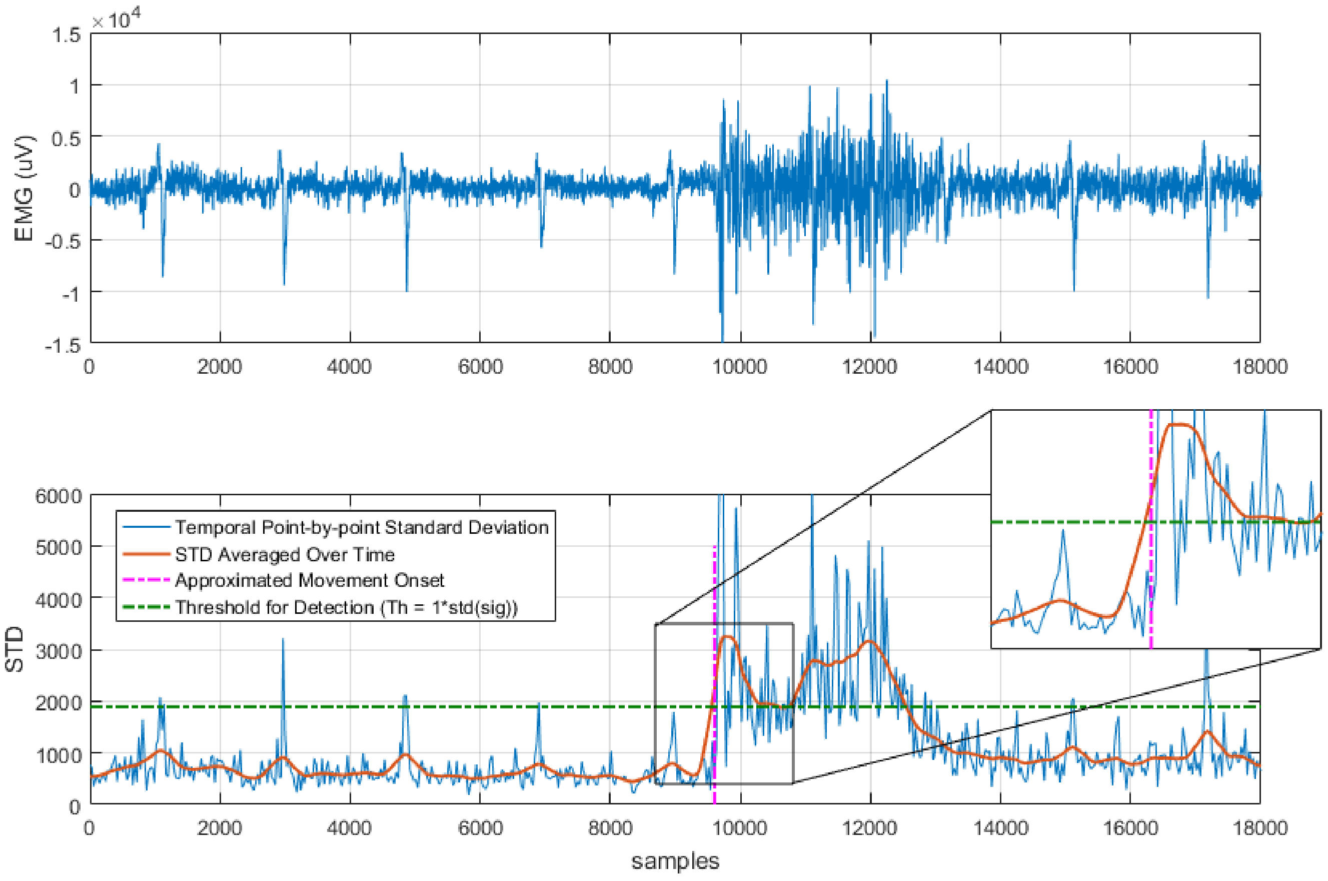

Figure 12: The implemented EMG movement onset detection procedure.

Inputs:

Input	Description
`trl_num`	number of trials
`elec_num`	number of electrodes used to record EEG
`emg_ch`	EMG channel of interest
`eeg_ch`	scalar or double vector of EEG channels of interest
`filename`	file name format as a string
`drift_flag`	baseline drift rejection flag, (options:`'drift'`, `'nodrift'`)
`onset_flag`	onset detection flag, (options: 1, 0)

Defaults:

Input	Default Values
`drift_flag`	`'drift'`
`onset_flag`	1

Outputs:

Output	Description
`eeg_data`	preprocessed EEGs of the selected channel from all trials
`emg_data`	preprocessed EMGs of the selected channel from all trials
`fs`	sampling frequency (Hz)
`emg_onset_sampl`	sample number of the movement onset
`emg_onset_time`	corresponding time of movement onset (Seconds)

Notes:

- An empty bracket [.] Must be assigned to not-specified values.
- EMG and EEG channels of interest have to be a doubles (i.e. either an integer or a vector of indices).
- `filename` needs to be defined in a way that `trl_num` can be used as an index to track and load all of the stored data. For instance, data file names may be defined as `filename_i` where $i = 1, 2, ..., N$ in which N represents the total number of data files. Refer to our sample data included in this toolbox to see another example of this, if needed.
- One should expect the `emg_onset_sampl` and `emg_onset_time` within output arguments in case they flagged the onset detection as 1 within input.

3.2 `trigger_synch.m`

`trigger_synch.m`

Purpose: Synchronizing EEG/EMG signals according to movement onset time.

Synopsis (global mode):

```
[ensemble_eeg, synch_eeg, trigger_time_sec, time_vec] = trigger_synch(eeg, fs,
                                onset_time)
```

Synopsis (local mode):

```
[ensemble_eeg, synch_eeg, trigger_time_sec, time_vec, synch_emg] = trigger_synch(eeg,
                          fs, onset_time, duration, emg)
```

Inputs:

Input	Description
`eeg`	cell array containing EEG channels of interest from all trials
`fs`	sampling frequency (Hz)
`onset_time`	vector of onset times (Seconds)
`duration`	required signal duration after movement onset (Seconds)
`emg`	cell array containing EMG channels of interest from all trials

Defaults:

Input	Default Values
`duration`	2
`emg`	{.}

Outputs:

Output	Description
`ensemble_eeg`	ensembles of EEG trials in a matrix
`synch_eeg`	synchronized EEG signals based on trigger time
`trigger_time_sec`	trigger onset flag (Seconds)
`time_vec`	time vector required for ERP plots
`synch_emg`	synchronized EMG signals based on trigger time

Notes:

- An empty bracket [.] Must be assigned to not-specified values.
- The provided function '`bdf2mat_main.m`' outputs the compatible EEG/EMG data for this function; however, you can simply store your data in a cell array in which each element includes one data trial(s) of interest.
- The output argument '`ensemble_eeg`' is an essential variable to be used within the provided function '`trigger_avg_erp.m`' and thus you can simply replace it with ˜ symbol if using this function independently.
- One should expect the `synch_emg` (e.g. synchronized EMG data) within output arguments in case they inputed EMG data.

3.3 `trigger_avg_erp.m`

`trigger_avg_erp.m`

Purpose: Trigger-averaged ERP time-course estimation function to extract the temporal dynamics of event-related potentials.

Synopsis (global mode):

```
[erp, synch_eeg, trigger_time_sec, time_vec] = trigger_avg_erp(eeg, fs, freq_band,
                                   onset_time)
```

Synopsis (local mode):

```
[erp, synch_eeg, trigger_time_sec, time_vec, synch_emg] = trigger_avg_erp(eeg, fs,
                     freq_band, onset_time, duration, emg)
```

Inputs:

Input	Description
`eeg`	cell array containing EEG channels of interest from all trias
`fs`	sampling frequency (Hz)
`freq_band`	frequency band of interest (refer to Notes below)
`onset_time`	vector of onset times (Seconds)
`duration`	required signal duration after movement onset (Seconds)
`emg`	cell array containing EMG channels of interest from all trials

Defaults:

Input	Default Values
`duration`	2
`emg`	{.}

Outputs:

Output	Description
`erp`	extracted ERP signal
`synch_eeg`	synchronized EEG signals based on trigger time
`trigger_time_sec`	trigger onset flag (Seconds)
`time_vec`	time vector required for ERP plots
`synch_emg`	synchronized EMG signals based on trigger time

Notes:

- An empty bracket [.] Must be assigned to not-specified values.
- Available options for `freq_band` are: '`delta`', '`theta`', '`alpha`', '`beta`' or '`gamma`' where are defined in ranges 1-4(Hz), 4-8(Hz), 8-12(Hz), 12-32(Hz) and 32-80Hz, respectively. In case you need to change these ranges, open up the function script and change `f0` and `bw` values for the band you wish to alter (i.e. lines 105 to 115). Note that `f0` and `bw` are center and bandwidth of the frequency range, respectively.
- The provided function '`bdf2mat_main.m`' outputs the compatible EEG/EMG data for this function; however, you can simply store your data in a cell array in which each element includes one data trial(s) of interest.
- One should expect the `synch_emg` (e.g. synchronized EMG data) within output arguments in case they inputed EMG data.

3.4 `trigger_avg_TF_erp.m`

`trigger_avg_TF_erp.m`

Purpose: Trigger-averaged ERP T/F representation to track the time-frequency dynamics of event-related potentials.

Synopsis (global mode):

```
[erp_tf, synch_eeg, trigger_time_sec, time_vec, freq_vec] = trigger_avg_TF_erp(eeg,
                                fs, onset_time)
```

Synopsis (local mode):

```
[erp_tf, synch_eeg, trigger_time_sec, time_vec, freq_vec, synch_emg] =
    trigger_avg_TF_erp(eeg, fs, onset_time, duration, method, emg)
```

Inputs:

Input	Description
`eeg`	cell array containing EEG channels of interest from all trials
`fs`	sampling frequency (Hz)
`onset_time`	vector of onset times (Seconds)
`duration`	required signal duration after movement onset (Seconds)
`method`	method for T/F representation, (options: '`STFT`', '`CWT`', '`NBCH`')
`emg`	cell array containing EMG channels of interest from all trials

Defaults:

Input	Default Values
`duration`	2
`method`	'`STFT`'
`emg`	{.}

Outputs:

Output	Description
`erp_tf`	estimated ERP time-frequency map
`synch_eeg`	synchronized eeg signals based on trigger time
`trigger_time_sec`	trigger onset flag (Seconds)
`time_vec`	time vector required for ERP plots
`freq_vec`	frequency vector required for ERP map plots
`synch_emg`	synchronized emg signals based on trigger time

Notes:

- An empty bracket [.] Must be assigned to not-specified values.
- The provided function '`bdf2mat_main.m`' outputs the compatible EEG/EMG data for this function; however, you can simply store your data in a cell array in which each element includes one data trial(s) of interest.
- One should expect the `synch_emg` (e.g. synchronized EMG data) within output arguments in case they inputed EMG data.

3.5 `erp_quantification.m`

`erp_quantification.m`

Purpose: ERP area (ERD and ERS events' area) quantification.

Synopsis (global mode):

```
[erd_area, ers_area, quant_erp] = erp_quantification(erp, fs, trigger_time)
```

Synopsis (local mode):

```
[erd_area, ers_area, quant_erp] = erp_quantification(erp, fs, trigger_time, ref_per,
                                        cof_intv)
```

Inputs:

Input	Description
`erp`	vector of estimated ERP signal
`fs`	sampling frequency (Hz)
`trigger_time`	trigger onset flag (Seconds)
`ref_per`	vector of reference period (refer to Notes below)
`cof_intv`	confidence interval coefficient

Defaults:

Input	Default Values
`ref_per`	[-1.3, -0.3] Seconds
`cof_intv`	3

Outputs:

Output	Description
`erd_area`	ERD events area
`ers_area`	ERS events area
`quant_erp`	ERP with quantified magnitude

Notes:

- An empty bracket [.] Must be assigned to not-specified values.
- Reference period variable '`ref_per`' have to be a double vector in [-a, -b] form where -a and -b are the edges of reference segment in Seconds. Minus sign shows that this period belongs to before movement onset (i.e. trigger time).
- `erd_area` and `ers_area` are stored in a variable-size cell array the size of which depend on the duration of ERP signal after movement onset. For instance, in figure 4 there are 2 ERD events and only one ERS event detected. As a results, the corresponding area cell arrays will be of size 1×2 and 1×1, respectively.

3.6 `TCPLV.m`

`TCPLV.m`

Purpose: PLV temporal dynamics estimated within 1 Second time-steps for any arbitrary time range before and after movement onset.

Synopsis (global mode):

```
tcplv = TCPLV(eeg, fs, onset_time)
```

Synopsis (local mode):

```
tcplv = TCPLV(eeg, fs, onset_time, freq_rng, duration, pairofint, pertnum)
```

Inputs:

Input	Description
`eeg`	cell array containing eeg channels of interest from all trials
`fs`	sampling frequency (Hz)
`onset_time`	vector of onset times (Seconds)
`freq_rng`	frequency range of interest (Hz)
`duration`	required temporal duration for tracking PLV dynamics (Seconds)
`pairofint`	channel pair of interest for PLV time-course
`pertnum`	number of perturbations while using the TFS phase estimation method

Defaults:

Input	Default Values
`freq_rng`	[12, 32] (Hz)
`duration`	[-3, 2] (Seconds)
`pairofint`	'all'
`pertnum`	100

Outputs:

Output	Description
`tcplv`	estimated time-course phase locking value (PLV) dynamics between eeg pairs of interest

Notes:

- An empty bracket [.] Must be assigned to not-specified values.
- `freq_rng` has to be in `[a, b]` form double vector where '`a`' and '`b`' are edges of the frequency band of interest in Hz.

- `duration` has to be in [-a, b] form double vector where '`a`' is the required duration (Seconds) prior to movement onset and '`b`' is the required duration (Seconds) after the movement onset. The specified duration will be segmented into 1-sec-long windows to calculate PLV. Refer to Section 2.2.5 for more details.
- `pairofint` can either be a double vector or '`all`' string. As a double vector it has to be in [a, b] form resulting in PLV measures between electrode number **a** and electrode number **b**. Moreover, if you use '`all`', a PLV *map* will be calculated between C3 electrode and all other electrodes to cover large-scale functional connectivity dynamics between motor cortex and other brain regions. In this case, C3 is chosen as the default electrode; however, if you wish to change this for your data, open up the script, scroll to line 143 of code and change the value 6 to you electrode number of interest.

3.7 `PWPLV.m`

`PWPLV.m`

Purpose: Pair-wise Phase-Locking Value (PLV) dynamics estimated within 1-Second time-steps for any arbitrary time range and all electrode pairs.

Synopsis (global mode):

```
pwplv = PWPLV(eeg, fs, onset_time)
```

Synopsis (local mode):

```
pwplv = PWPLV(eeg, fs, onset_time, freq_rng, duration, pertnum, plot_flag)
```

Inputs:

Input	Description
`eeg`	cell array containing eeg channels of interest from all trials
`fs`	sampling frequency (Hz)
`onset_time`	vector of onset times (Seconds)
`freq_rng`	frequency range of interest (Hz)
`duration`	required temporal duration for tracking PLV dynamics (Seconds)
`pertnum`	number of perturbations while using the TFS phase estimation method
`plot_flag`	flag to visualize the results or not (options: '`plot`', '`noplot`')

Defaults:

Input	Default Values
`freq_rng`	[12, 32] (Hz)
`duration`	[-3, 2] (Seconds)
`pertnum`	100
`plot_flag`	'`plot`'

Outputs:

Output	Description
`pwplv`	estimated pairwise phase locking value (PLV) between all possible electrode pairs

Notes:

- An empty bracket [.] Must be assigned to not-specified values.
- `freq_rng` has to be in `[a, b]` form double vector where '`a`' and '`b`' are edges of the frequency band of interest in Hz.
- `duration` has to be in [-a, b] form double vector where '`a`' is the required duration (Seconds) prior to movement onset and '`b`' is the required duration (Seconds) after the movement onset. The specified duration will be segmented into 1-sec-long windows to calculate PLV. Refer to Section 2.2.5 for more details.
- This function utilizes all of the electrodes (i.e. not just motor cortex electrode C3) and all possible combinations to calculate PLV values and generate connectivity maps. Refer to Section 2.2.6 for more details.

3.8 `PWCoherence.m`

`PWCoherence.m`

Purpose: Pair-wise Magnitude-Squared Coherence (MSC) dynamics estimated within 1-Second time-steps for any arbitrary time range and all electrode pairs.

Synopsis (global mode):

```
pwcoher = PWCoherence(eeg, fs, onset_time)
```

Synopsis (local mode):

```
pwcoher = PWCoherence(eeg, fs, onset_time, freq_rng, duration, plot_flag)
```

Inputs:

Input	Description
`eeg`	cell array containing eeg channels of interest from all trials
`fs`	sampling frequency (Hz)
`onset_time`	vector of onset times (Seconds)
`freq_rng`	frequency range of interest (Hz)
`duration`	required temporal duration for tracking PLV dynamics (Seconds)
`plot_flag`	flag to visualize the results or not (options: '`plot`', '`noplot`')

Defaults:

Input	Default Values
`freq_rng`	[12, 32] (Hz)
`duration`	[-3, 2] (Seconds)
`plot_flag`	'plot'

Outputs:

Output	Description
`pwcoher`	estimated pairwise magnitude squared coherence (MSC) between all possible electrode pairs

Notes:

- An empty bracket [.] Must be assigned to not-specified values.
- `freq_rng` has to be in `[a, b]` form double vector where '`a`' and '`b`' are edges of the frequency band of interest in Hz.
- `duration` has to be in [-a, b] form double vector where '`a`' is the required duration (Seconds) prior to movement onset and '`b`' is the required duration (Seconds) after the movement onset. The specified duration will be segmented into 1-sec-long windows to calculate PLV. Refer to Section 2.2.5 for more details.
- This function utilizes all of the electrodes (i.e. not just motor cortex electrode C3) and all possible combinations to calculate MSC values and generate connectivity maps. Refer to Section 2.2.6 for more details.

3.9 emg_quantification.m

```
emg_quantification.m
```

Purpose: EMG signal analysis and quantification.

Synopsis (global mode):

```
[emg_quant, synch_emg2, ecg_estimate2_bl, time_vec] = emg_quantification(emg_data,
                                 fs, emg_onset_sampl)
```

Synopsis (local mode):

```
[emg_quant, synch_emg2, ecg_estimate2_bl, time_vec] = emg_quantification(emg_data,
                          fs, emg_onset_sampl, duration)
```

Inputs:

Input	Description
`emg_data`	cell array containing EMG signals of all trials
`fs`	sampling frequency (Hz)
`emg_onset_sampl`	vector containing movement onset samples of all trials
`duration`	duration of signal required after onset

Defaults:

Input	Default Values
`duration`	2

Outputs:

Output	Description
`emg_quant`	vector containing quantified EMG values
`synch_emg2`	cell array containing synchronized EMG trials based on movement onset
`ecg_estimate2_bl`	extracted ECG signal from EMG channels
`time_vec`	time-vector required for plotting quantified EMG

3.10 `drift_reject.m`

`drift_reject.m`

Purpose: Drift (baseline wander) cancellation from biosignal recordings.

Synopsis (global mode):

```
sig = drift_reject(raw_sig, L1)
```

Synopsis (local mode):

```
sig = drift_reject(raw_sig, L1, L2, approach)
```

Inputs:

Input	Description
`raw_sig`	matrix or vector of raw recordings
`L1`	first stage window length (sample)
`L2`	second stage window length (sample)
`approach`	filtering approach, (options: 'md' or 'mn')

Defaults:

Input	Default Values
`L2`	L1 (sample)
`approach`	'mn'

Outputs:

Output	Description
`sig`	matrix or vector of drift-rejected signals

Notes:

- An empty bracket [.] Must be assigned to not-specified values.

3.11 `emg_onset.m`

`emg_onset.m`

Purpose: EMG onset detection function based on introduced two-stage thresholding approach in Section 2.2.7.

Synopsis (global mode):

```
[onset_sampl, onset_time] = emg_onset(emg, fs, W)
```

Synopsis (local mode):

```
[onset_sampl, onset_time] = emg_onset(emg, fs, W, th_coeff, Trl)
```

Inputs:

Input	Description
`emg`	the emg signal
`fs`	EMG signal's sampling frequency (Hz)
`W`	window length for STD calculation (sample)
`th_coeff`	threshold coeff for onset detection
`Trl`	current trial number

Defaults:

Input	Default Values
`th_coeff`	1
`Trl`	—

Outputs:

Output	Description
`onset_sampl`	sample number of the movement onset
`onset_time`	corresponding time of movement onset

Notes:

- `th_coeff` is a coefficient that will be multiplied by the standard deviation of the EMG baseline and by default is set to 1 so that the estimation threshold will be equal to one-standard-deviation of baseline (i.e. 1×STD{emg}).
- An empty bracket [.] Must be assigned to not-specified values.

3.12 `phase_est.m`

`phase_est.m`

Purpose: Instantaneous phase estimation by the Transfer Function Perturbation (TFP) method [33, 34, 13].

Synopsis (global mode):

```
[phase_avg, freq_avg, amp_avg, analytic_sig_avg] = phase_est(sig, fs, f0, bw_base)
```

Synopsis (local mode):

```
[phase_avg, freq_avg, amp_avg, analytic_sig_avg] = phase_est(sig, fs, f0, bw_base,
                                        pertnum)
```

Inputs:

Input	Description
`sig`	input raw signal
`fs`	sampling frequency (Hz)
`f0`	center frequency of the passband (Hz)
`bw_base`	bandwidth of the frequency filter (Hz)
`pertnum`	number of perturbations while using the TFP phase estimation method

Defaults:

Input	Default Values
`pertnum`	100

Outputs:

Output	Description
`phase_avg`	estimated instantaneous phase of input signal
`freq_avg`	estimated instantaneous frequency of input signal
`amp_avg`	estimated instantaneous envelope of input signal
`analytic_sig_avg`	generated analytic form of input signal

Notes:

- `pertnum` is by default set to 100, however the value mostly depends on the application and also the required level of reliability. Nevertheless, 100 has been tested in several applications before and is considered enough [34, 27, 25, 26].
- TFP perturbation parameters are set according to the findings of [33] and are chosen in a way that are physiologically irrelevant. In case one needs to change these values due to different application specs, refer to lines 83 to 88 of code.
- An empty bracket [.] Must be assigned to not-specified values.

3.13 `PLV_PhaseSeq.m`

`PLV_PhaseSeq.m`

Purpose: Calculating Phase Locking Value (PLV) matrix (Pairwise PLV) using phase sequences [13].

Synopsis:

```
PLV = PLV_PhaseSeq(phase_sig)
PLV = PLV_PhaseSeq(phase_sig1, phase_sig2, phase_sig3, ...)
```

Inputs:

Input	Description
`phase_sig`	input phase matrix
`phase_sig1`	input phase vector #1
`phase_sig2`	input phase vector #2
•	•
•	•

Outputs:

Output	Description
`PLV`	Pairwise PLV matrix

Notes:

- While using the first case, the `phase_sig` have to be a matrix with at least two rows where each row represents a phase signal. In second case, each of the `phase_sig1...phase_sign` are row vectors of phase sequences. This option is provided in case that someone needs to calculate the PLV matrix between separate phase signals.

3.14 sig_trend.m

`sig_trend.m`

Purpose: Calculating the trend of a signal using local minima.

Synopsis:

```
[Tr_sig, loc] = sig_trend(sig)
```

Inputs:

Input	Description
`sig`	input raw signal

Outputs:

Output	Description
`Tr_sig`	trend vector
`loc`	returns the locations required for plotting the trend

3.15 task_separator.m

`task_separator.m`

Purpose: Task based separation of data to avoid memory overuse.

Synopsis (global mode):

```
sep_file_names = task_separator(filename)
```

Inputs:

Input	Description
`filename`	original whole-data file name as a string

Outputs:

Output	Description
`sep_file_names`	cell array containing separated files' names

4 Acknowledgment

The authors would like to thank Dr. Maysam Ghovanloo[2], Dr. Minoru Shinohara[3], Dr. Boris I. Prilutsky[4], Dr. Andrew J. Butler[5] and Zhenxuan Zhang[6] for their insightful discussions and comments.

5 References

References

[1] G. Pfurtscheller and F. L. Da Silva, "Event-related eeg/meg synchronization and desynchronization: basic principles," *Clinical neurophysiology*, vol. 110, no. 11, pp. 1842–1857, 1999.

[2] T. C. Handy, *Event-related potentials: A methods handbook.* MIT press, 2005.

[3] S. J. Luck, *An introduction to the event-related potential technique.* MIT press, 2014.

[4] S. Makeig, S. Debener, J. Onton, and A. Delorme, "Mining event-related brain dynamics," *Trends in cognitive sciences*, vol. 8, no. 5, pp. 204–210, 2004.

[5] M. D. Greicius, B. Krasnow, A. L. Reiss, and V. Menon, "Functional connectivity in the resting brain: a network analysis of the default mode hypothesis," *Proceedings of the National Academy of Sciences*, vol. 100, no. 1, pp. 253–258, 2003.

[6] G. Carter, C. Knapp, and A. Nuttall, "Estimation of the magnitude-squared coherence function via overlapped fast fourier transform processing," *IEEE transactions on audio and electroacoustics*, vol. 21, no. 4, pp. 337–344, 1973.

[7] J.-P. Lachaux, E. Rodriguez, J. Martinerie, and F. J. Varela, "Measuring phase synchrony in brain signals," *Human brain mapping*, vol. 8, no. 4, pp. 194–208, 1999.

[8] F. Varela, J.-P. Lachaux, E. Rodriguez, and J. Martinerie, "The brainweb: phase synchronization and large-scale integration," *Nature reviews neuroscience*, vol. 2, no. 4, p. 229, 2001.

[9] M. G. Rosenblum, A. S. Pikovsky, and J. Kurths, "Phase synchronization of chaotic oscillators," *Physical review letters*, vol. 76, no. 11, p. 1804, 1996.

[10] A. L. Ricamato and J. M. Hidler, "Quantification of the dynamic properties of emg patterns during gait," *Journal of electromyography and kinesiology*, vol. 15, no. 4, pp. 384–392, 2005.

[11] C. B. Walter, "Temporal quantification of electromyography with reference to motor control research," *Human Movement Science*, vol. 3, no. 1-2, pp. 155–162, 1984.

[12] R. Sameni, "The open-source electrophysiological toolbox (oset), version 3.1, 2014," *URL http://www. oset. ir*, 2014.

[13] E. Seraj, "Cerebral signal phase analysis toolbox–user guide," *arXiv preprint arXiv:1610.02249*, 2016.

[2]GT-Bionics Lab, School of Electrical and Computer Engineering, Georgia Tech, Atlanta, GA, USA
[3]Human Neuromuscular Physiology Lab, School of Biological Sciences, Georgia Tech, Atlanta, GA, USA
[4]Biomechanics and Motor Control Lab, School of Biological Sciences, Georgia Tech, Atlanta, GA, USA
[5]B.F. Lewis School of Nursing & Health Professions, Georgia State University, Atlanta, GA, USA
[6]GT-Bionics Lab, School of Electrical and Computer Engineering, Georgia Tech, Atlanta, GA, USA

[14] G. Tcheslavski, "eeg bdf reader matlab function." [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/13070-eeg-bdf-reader?focused=5083421&tab=function

[15] B. BioSemi, "Biosemi activetwo.[eeg system]," *Amsterdam: BioSemi*, 2011.

[16] L. Smith, "Activetwo system operating guidelines," *Cortech Solutions, Inc.*, 2009.

[17] T. C. Technologies, "10/20 system positioning manual." 2012.

[18] G. Repovs, "Dealing with noise in eeg recording and data analysis," in *Informatica Medica Slovenica*, vol. 15, no. 1, 2010, pp. 18–25.

[19] G. A. Light, L. E. Williams, F. Minow, J. Sprock, A. Rissling, R. Sharp, N. R. Swerdlow, and D. L. Braff, "Electroencephalography (eeg) and event-related potentials (erps) with human participants," *Current protocols in neuroscience*, vol. 52, no. 1, pp. 6–25, 2010.

[20] J. B. Nitschke, G. A. Miller, and E. W. Cook, "Digital filtering in eeg/erp analysis: Some technical and empirical comparisons," *Behavior Research Methods, Instruments, & Computers*, vol. 30, no. 1, pp. 54–67, 1998.

[21] R. Lyons, "Understanding cascaded integrator-comb filters," *Embed Syst Program*, vol. 18, no. 4, pp. 14–27, 2005.

[22] G. Pfurtscheller, "Graphical display and statistical evaluation of event-related desynchronization (erd)," *Electroencephalography and clinical neurophysiology*, vol. 43, no. 5, pp. 757–760, 1977.

[23] B. Graimann and G. Pfurtscheller, "Quantification and visualization of event-related changes in oscillatory brain activity in the time–frequency domain," *Progress in brain research*, vol. 159, pp. 79–97, 2006.

[24] L. Cohen, *Time-frequency analysis.* Prentice hall, 1995, vol. 778.

[25] E. Seraj and F. Karimzadeh, "Improved detection rate in motor imagery based bci systems using combination of robust analytic phase and envelope features," in *Electrical Engineering (ICEE), 2017 Iranian Conference on.* IEEE, 2017, pp. 24–28.

[26] E. Seraj, "An investigation on the utility and reliability of electroencephalogram phase signal upon interpreting cognitive responses in the brain: A critical discussion," *Journal of Advanced Medical Sciences and Applied Technologies*, vol. 2, no. 4, pp. 299–312, 2017.

[27] F. Karimzadeh, R. Boostani, E. Seraj, and R. Sameni, "A distributed classification procedure for automatic sleep stage scoring based on instantaneous electroencephalogram phase and envelope features," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, no. 2, pp. 362–370, 2018.

[28] E. Seraj, M. Yazdi, and N. Shahparian, "fmri based cerebral instantaneous parameters for automatic alzheimer's, mild cognitive impairment and healthy subject classification," *arXiv preprint arXiv:1904.07441*, 2019.

[29] R. Boostani, F. Karimzadeh, and M. Nami, "A comparative review on sleep stage classification methods in patients and healthy individuals," *Computer methods and programs in biomedicine*, vol. 140, pp. 77–91, 2017.

[30] F. Karimzadeh, E. Seraj, R. Boostani, and M. Torabi-Nami, "Presenting efficient features for automatic cap detection in sleep eeg signals," in *2015 38th International Conference on Telecommunications and Signal Processing (TSP).* IEEE, 2015, pp. 448–452.

[31] F. Karimzadeh, M. Nami, and R. Boostani, "Sleep microstructure dynamics and neurocognitive performance in obstructive sleep apnea syndrome patients," *Journal of integrative neuroscience*, vol. 16, no. 2, pp. 127–142, 2017.

[32] E. Seraj, "Cerebral synchrony assessment: A general review on cerebral signals' synchronization estimation concepts and methods," *arXiv preprint arXiv:1612.04295*, 2016.

[33] R. Sameni and E. Seraj, "A robust statistical framework for instantaneous electroencephalogram phase and frequency estimation and analysis," *Physiological measurement*, vol. 38, no. 12, p. 2141, 2017.

[34] E. Seraj and R. Sameni, "Robust electroencephalogram phase estimation with applications in brain-computer interface systems," *Physiological measurement*, vol. 38, no. 3, p. 501, 2017.

[35] F. T. Sun, L. M. Miller, and M. D'esposito, "Measuring interregional functional connectivity using coherence and partial coherence analyses of fmri data," *Neuroimage*, vol. 21, no. 2, pp. 647–658, 2004.

[36] J. Groß, J. Kujala, M. Hämäläinen, L. Timmermann, A. Schnitzler, and R. Salmelin, "Dynamic imaging of coherent sources: studying neural interactions in the human brain," *Proceedings of the National Academy of Sciences*, vol. 98, no. 2, pp. 694–699, 2001.

[37] N. W. Willigenburg, A. Daffertshofer, I. Kingma, and J. H. van Dieën, "Removing ecg contamination from emg recordings: A comparison of ica-based and other filtering procedures," *Journal of electromyography and kinesiology*, vol. 22, no. 3, pp. 485–493, 2012.

[38] J. D. Drake and J. P. Callaghan, "Elimination of electrocardiogram contamination from electromyogram signals: An evaluation of currently used removal techniques," *Journal of electromyography and kinesiology*, vol. 16, no. 2, pp. 175–187, 2006.

[39] K. Aminian, C. Ruffieux, and P. Robert, "Filtering by adaptive sampling (fas)," *Medical and Biological Engineering and Computing*, vol. 26, no. 6, pp. 658–662, 1988.

[40] S. Abbaspour and A. Fallah, "Removing ecg artifact from the surface emg signal using adaptive subtraction technique," *Journal of biomedical physics & engineering*, vol. 4, no. 1, p. 33, 2014.

[41] F. Nougarou, D. Massicotte, and M. Descarreaux, "Efficient procedure to remove ecg from semg with limited deteriorations: Extraction, quasi-periodic detection and cancellation," *Biomedical Signal Processing and Control*, vol. 39, pp. 1–10, 2018.

[42] S. Abbaspour, M. Lindén, and H. Gholamhosseini, "Ecg artifact removal from surface emg signal using an automated method based on wavelet-ica." in *pHealth*, 2015, pp. 91–97.

[43] M. Chen, X. Zhang, X. Chen, M. Zhu, G. Li, and P. Zhou, "Fastica peel-off for ecg interference removal from surface emg," *Biomedical engineering online*, vol. 15, no. 1, p. 65, 2016.

[44] Y. Li, X. Chen, X. Zhang, and P. Zhou, "Ecg artifact removal from emg recordings using independent component analysis and adapted filter," in *Neural Engineering (NER), 2013 6th International IEEE/EMBS Conference on.* IEEE, 2013, pp. 347–350.

[45] S. Akselrod, D. Gordon, F. A. Ubel, D. C. Shannon, A. Berger, and R. J. Cohen, "Power spectrum analysis of heart rate fluctuation: a quantitative probe of beat-to-beat cardiovascular control," *science*, vol. 213, no. 4504, pp. 220–222, 1981.

[46] F. Barlaam, M. Descoins, O. Bertrand, T. Hasbroucq, F. Vidal, C. Assaiante, and C. Schmitz, "Time–frequency and erp analyses of eeg to characterize anticipatory postural adjustments in a bimanual load-lifting task," *Frontiers in human neuroscience*, vol. 5, p. 163, 2011.

[47] O. E. Krigolson, “Event-related brain potentials and the study of reward processing: Methodological considerations,” *International Journal of Psychophysiology*, 2017.

[48] P. R. Cavanagh and P. V. Komi, “Electromechanical delay in human skeletal muscle under concentric and eccentric contractions,” *European journal of applied physiology and occupational physiology*, vol. 42, no. 3, pp. 159–163, 1979.

[49] P. Bonato, T. D’Alessio, and M. Knaflitz, “A statistical method for the measurement of muscle activation intervals from surface myoelectric signal during gait,” *IEEE Transactions on biomedical engineering*, vol. 45, no. 3, pp. 287–299, 1998.

[50] J. Drapała, K. Brzostowski, A. Szpala, and A. Rutkowska-Kucharska, “Two stage emg onset detection method,” *Archives of Control Sciences*, vol. 22, no. 4, pp. 427–440, 2012.