

RIPPLELAB USER'S GUIDE

Prepared by: Miguel Navarrete and Mario Valderrama
February 25, 2016

Revision Sheet

Release No	Date	Revision Description
1.1	25/02/2016	Inclusion of load data warnings and RIPPLELAB Script section



Universidad de
los Andes



Table of Contents

Revision Sheet	i
1. GENERAL INFORMATION.....	1
1.1. System Overview	1
1.2. Organization of the Manual.....	1
2. SYSTEM SUMMARY	2
2.1. System configuration.....	2
3. GETTING STARTED.....	2
3.1. Installing the software	2
4. USING RIPPLELAB	3
4.1. Importing Files	5
4.1.1. Including a new file	5
4.1.2. Removing a File	5
4.1.3. About the customized MATLAB Files	6
4.2. Channel Selection	7
4.2.1. Montage selection	8
4.3. Channel's Load.....	9
4.4. Data pre-processing.....	10
4.4.1. Filtering	11
4.4.2. Time Frequency analysis	13
4.4.3. Cursors measures.....	14
4.4.4. Power spectrum	15
5. HFOs DETECTION.....	15
5.1. General Controls.....	15
5.2. Manual detection of HFOs.....	16
5.3. Automatic detection of HFOs	18
5.4. HFO detection process and logs	20
6. HFOs VALIDATION	21
6.1. Load a HFO analysis	23
6.2. Review a HFO analysis	23
6.3. Save a HFO analysis.....	24

7.	RIPPLELAB SCRIPTS.....	24
7.1.	Scripts overview.....	24
7.2.	Including a new EEG format.....	25
7.3.	Including a new detection method.....	27
8.	Appendixes.....	29
8.1.	GNU GENERAL PUBLIC LICENSE	29
9.	REFERENCES	33

Table of figures

FIGURE 1.	STEPS FOR HFO DETECTION USING THE RIPPLELAB SOFTWARE	3
FIGURE 2.	RIPPLELAB'S MAIN WINDOW	4
FIGURE 3.	SELECT CHANNELS WINDOW	6
FIGURE 4.	CONVERT *.MAT FILES WINDOW.....	7
FIGURE 5.	CHANNEL SELECTION IN THE SELECT CHANNELS WINDOW	8
FIGURE 6.	ASSEMBLY CHANNELS WINDOW OPTIONS	8
FIGURE 7.	FILTER TOOLS	11
FIGURE 8.	TIME-FREQUENCY TOOLS.....	13
FIGURE 9.	CURSOR AND SPECTRUM TOOLS	14
FIGURE 10.	THE PSD PANEL.....	15
FIGURE 11.	HFO-DETECTION METHODS WINDOW.....	16
FIGURE 12.	RIPPLELAB'S MAIN FIGURE SET TO DEVELOP THE HFO VISUAL MARKING	17
FIGURE 13.	AUTOMATIC DETECTION PARAMETERS.....	19
FIGURE 14.	RUNNING HFO DETECTION WINDOW.....	20
FIGURE 15.	HFO ANALYSIS TOOL INTERFACE.....	21

1. GENERAL INFORMATION

RIPPLELAB is a multi-window GUI developed in MATLAB for the analysis of high frequency oscillations. It is intended to be a user-friendly and intuitive tool, where users with technical and non-technical backgrounds can explore and analyze brain oscillations from different types of electrophysiological data, especially at high frequency ranges.

Questions and suggestions should be addressed to:

Miguel Navarrete mnavarrete@gmail.com or Mario Valderrama mvalderm@gmail.com



1.1. System Overview

RIPPLELAB is an MATLAB application, which allows users the manual and automatic detection and validation of cortical High Frequency Oscillations (HFOs). This application loads data from different EEG formats to process, analyze and display HFOs. Its operational status is under development. RIPPLELAB operates on Windows, OS and Linux where MATLAB R2012b or later is installed.

1.2. Organization of the Manual

The user's manual consists of eight sections: General Information, System Summary, Getting Started, Using RIPPLELAB, HFOs detection, HFOs Validation, RIPPLELAB Scripts, Appendixes and References.

General Information section explains in general terms the system and the purpose for which it is intended. System Summary section provides a general overview of the system. The summary outlines the uses of RIPPLELAB's software requirements. Getting Started section explains how to get RIPPLELAB and the installation process. Using RIPPLELAB section provides a detailed description of the software functions for channel load, display and pre-process. HFOs detection section provides a detailed description of the RIPPLELAB's options to detect HFOs. HFOs detection section provides a detailed description of the RIPPLELAB's options to validate HFOs. RIPPLELAB Scripts provides a general description of how the software scripts are organized, and how some features can be modified according to the user requirements. Finally, the Appendixes section presents some supplemental information, and References section presents the cited bibliography.

In this user's manual, the warning icon  indicates information of processes that requires special conditions to operate. The info icon  highlights important information about the functions described in this document.

2. SYSTEM SUMMARY

RIPPLELAB has been released under GNU Public License version 3, and the source code and documentation can be found in <https://github.com/BSP-Uniandes/RIPPLELAB/>.

2.1. System configuration

RIPPLELAB was developed exclusively using MATLAB scripts, so the compilation of native libraries or external functions is not required. This multi-platform tool can be used on OS X, Linux and Windows 32 and 64-bit architectures, and it can be installed as a MATLAB App in versions equal or later to R2012b.

3. GETTING STARTED

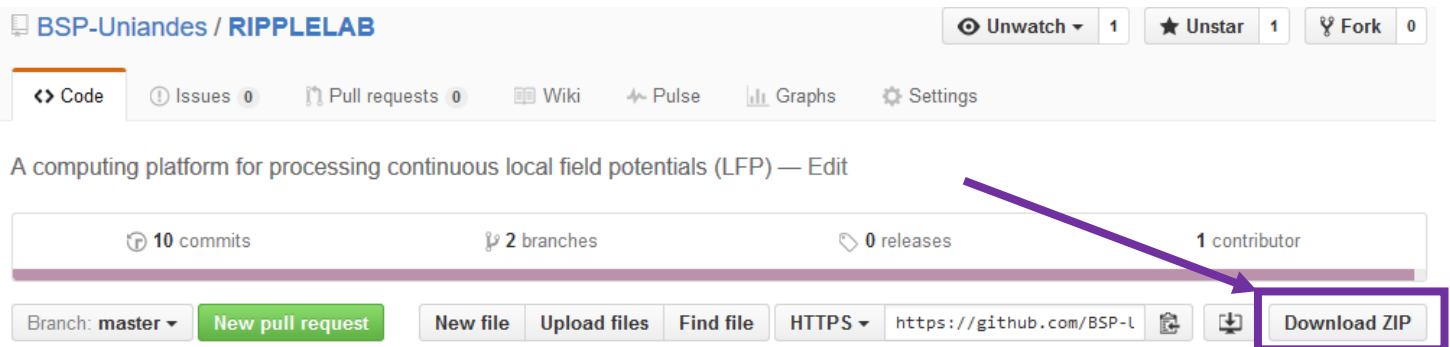
RIPPLELAB was developed in MATLAB version 7.12 (R2011a), and it has been tested in versions 8 (R2012b) and 8.5 (R2015a). To reduce dependency of proprietary MATLAB toolboxes, we integrated some low level functions from BioSig (Vidaurre et al. 2011) and FiledTrip (Oostenveld et al. 2011) toolboxes for specific signal processing and statistical analysis tasks. Even so, if present, RIPPLELAB identifies the MATLAB Signal Processing Toolbox and implements it in order to achieve faster signal processing algorithms.

All the RIPPLELAB images in this manual correspond to the GUI display using MATLAB 8.5 (R2015a)

3.1. Installing the software

RIPPLELAB scripts are found in <https://github.com/BSP-Uniandes/RIPPLELAB/>.

- In the GitHub webpage, please select the [Download ZIP] option:



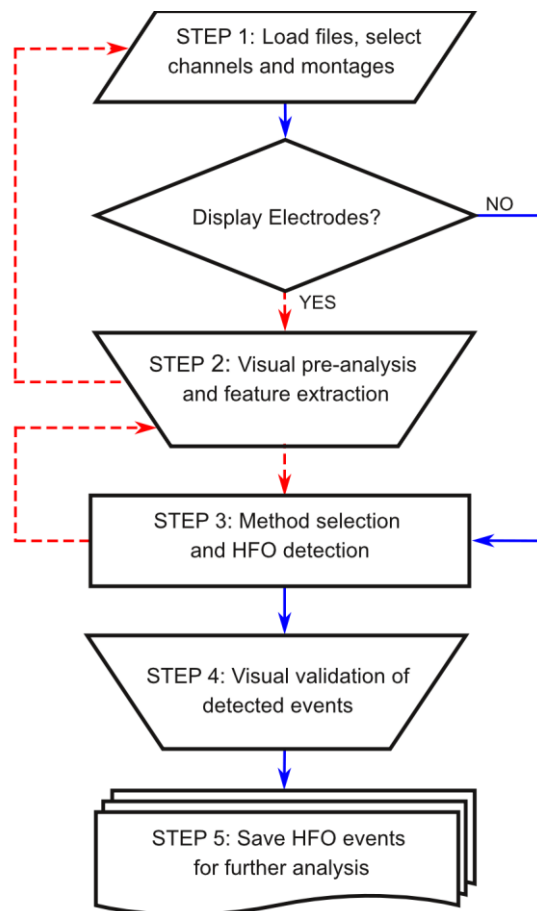
- Select a location to unzip the RIPPLELAB files.
- If you wish, you can add the RIPPLELAB files to the MATLAB path: [Home> Set Path> Add with subfolders]
- Start to use RIPPLELAB

4. USING RIPPLELAB

The complete procedure for detection and analysis of HFOs through RIPPLELAB consists of several steps that are presented in the following subsections and are summarized in **Fig 1**. The steps to use the RIPPLELAB interface are:

- Step 1: Import Files to read
- Step 2: Pre-processing
- Step 3: HFO detection
- Step 4: HFO validation by visual inspection
- Step 5: Save analysis

Figure 1. Steps for HFO detection using the RIPPLELAB software

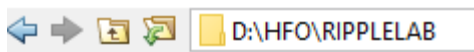


To open RIPPLELAB main GUI, several options are available:

- a. If RIPPLELAB files are in the MATLAB path, write the script name on the workspace.

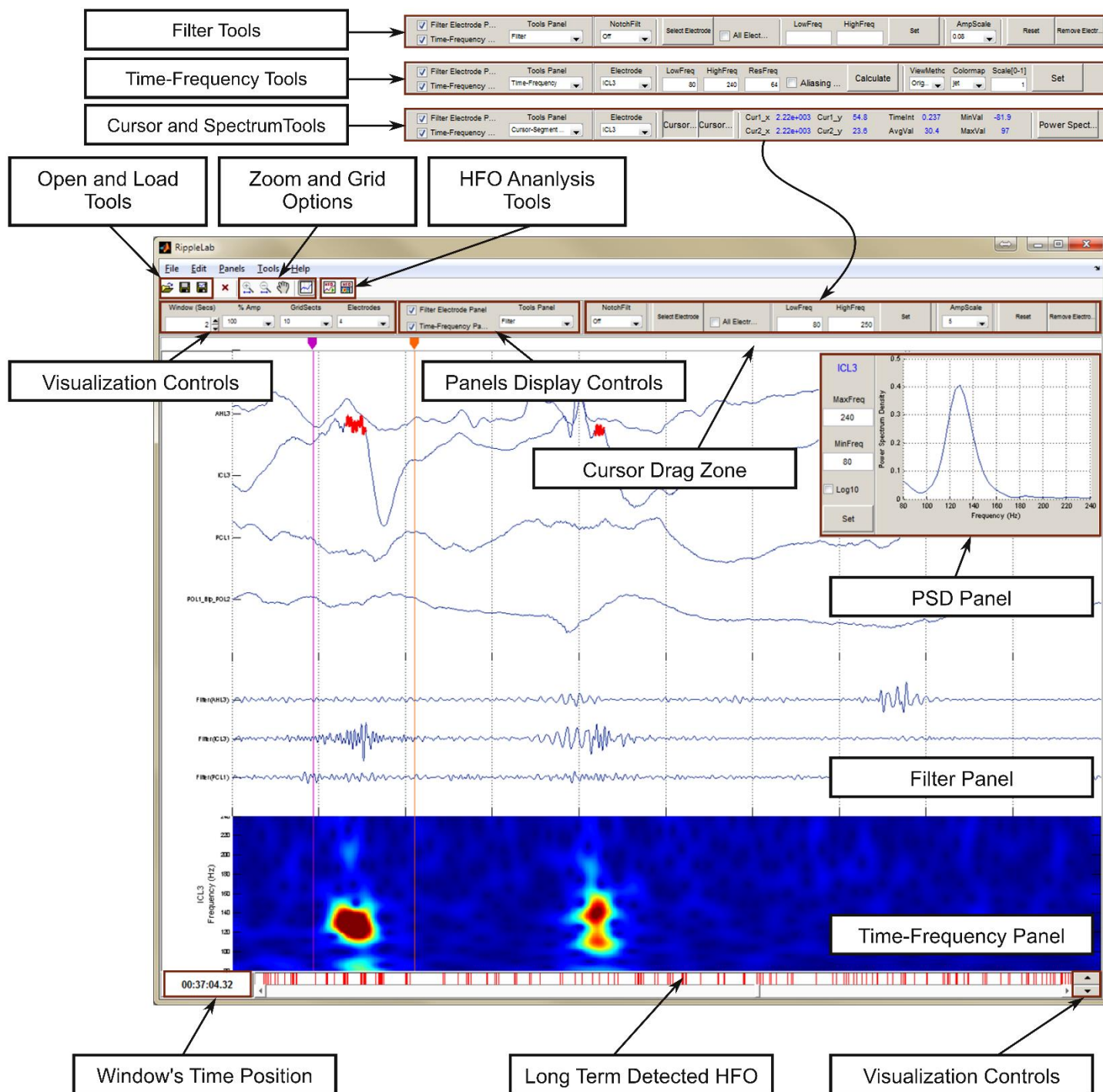
Command Window
`p_RippleLab`

- b. If files are not in the MATLAB path, you must select the RIPPLELAB location on the MATLAB current Folder. For instance:



- c. Starting the application will bring the main window figure (Fig 2).

Figure 2. RIPPLELAB's main window




4.1. Importing Files

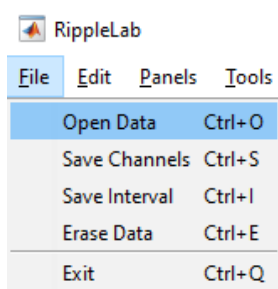
The user can select one or several files for the analysis of long-term recordings. Available and tested EEG formats are:

- Generic standard formats (*.rec, *.edf)
- EPILEPSIAE file format (*.data)
- Nicolet Files (*.eeg)
- Neuralynx Files (*.ncs)
- Plexon Files (*.plx)
- Axon binary Files (*.abf)
- Customized MATLAB Files (*.mat)
- EEGLAB files (*.set)

Furthermore, RIPPLELAB includes the open source FieldTrip (Oostenveld et al. 2011) scripts to open EEG formats. However, not all the formats listed by FieldTrip have been tested.

4.1.1. Including a new file

- To include a new file to RIPPLELAB select the Open Data icon in the toolbar menu  or go to File > Open Data on the RIPPLELAB's main figure [Fig 2: Open and Load Tools Box]:



- Then, the Select Channels Window is displayed (Fig 3)
- Go to the *+Files* button in the Select Channels Window
- Select the file or files to open. Only select files of the same format when selecting multiple files.



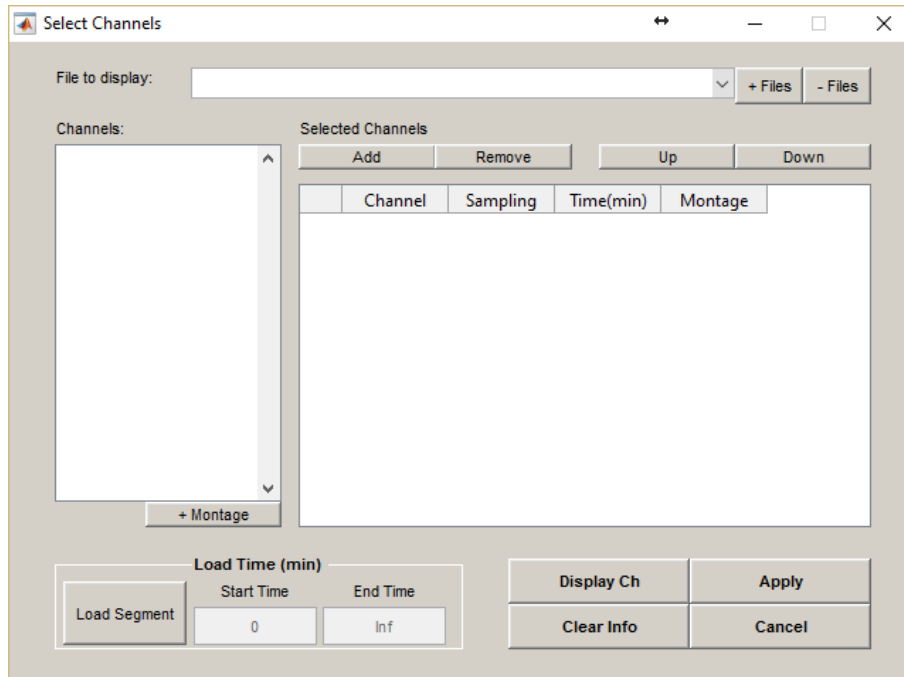
When more than one file has been selected, only the common electrodes will be visible. If there is not common electrodes, the *Channels* list will be empty.

- The selected file will be displayed in the *File to display* popup menu in the Select Channels Windows


4.1.2. Removing a File

- If multiple files have been selected and you want to remove one file of the list, then click on the *-Files* button in the Select Channels Window and select the file to remove.
- If you want to remove all the files or the file selected, then click the *Clear Info* button in the Select Channels Window.

Figure 3. Select Channels window



4.1.3. About the customized MATLAB Files

 RIPPLELAB have the possibility to open customized MAT files, but this files need to be organized in the appropriate structure to be correctly opened.

The correct File MAT file structure required by RIPPLELAB to be opened is:

- The custom MATLAB File (*.mat) for RIPPELAB is a structure with two fields: Header and Data.
- The **Data** field is an nxm matrix with n samples and m channels.
- **Header** is another structure with four fields:
 - **Sampling**, a vector of sampling rates in Hz for each channel,
 - **IniTime** which is a vector of the initial time of the register ([HH mm SS.ms]),
 - **Samples** representing the number of samples in a channel,
 - **Labels** which is a cell vector of strings with the channel names.

Nevertheless, when a MAT file with different file structure is selected, then the *Convert *.mat files* window is opened to convert MAT files to the appropriate format (**Fig 4**).

In this window, you can select the variable corresponding to data, scale, sampling rate and labels. Also, you can add this information manually for the scale, sampling rate, labels and start time. Each parameter includes a help button for specific information.

After selecting the appropriate parameters, three options are available:

- **Load MAT file Button:** When this option is selected, the data is loaded in the *Select Channels* window once. However, the file structure is not retained for future sessions. Thus, the next time RIPPLELAB opens this file, the file parameters must be introduced again.
- **Save MAT file Button:** When this option is selected, the system ask to save a new file with the customized MAT file structure. This option do not load the data info into the *Select Channels* window, and it must be selected as a new file again. However, the file structure is retained for future sessions. Thus, the next time RIPPLELAB opens this file, the file parameters do not must be introduced again.
- **Cancel Button:** This option ignores all the parameters and returns to the previous window.

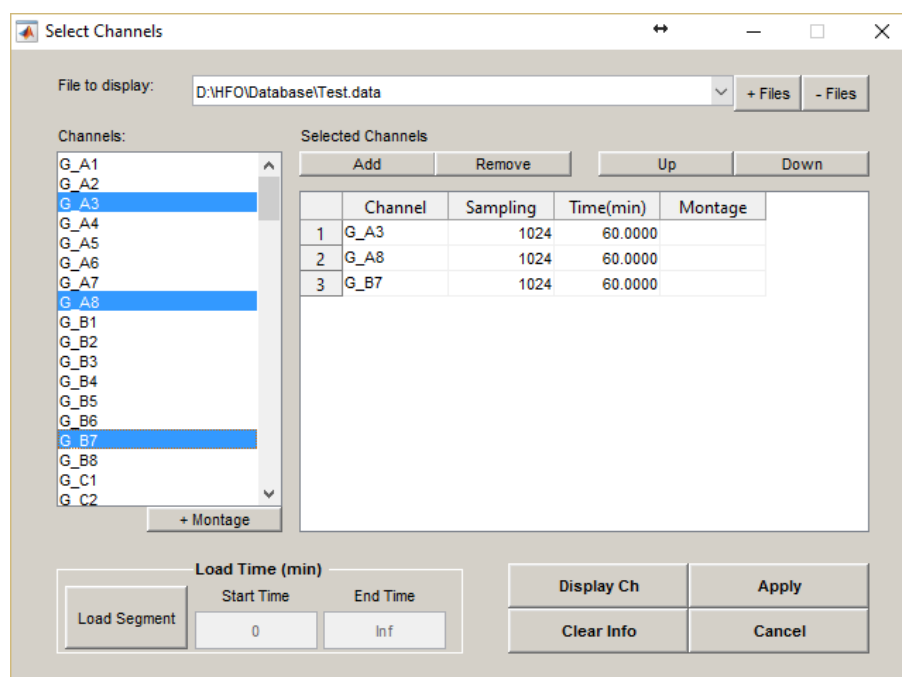
Figure 4. Convert *.mat files window

4.2. Channel Selection

After file selection, the respective electrodes are going to be presented in the *Channels* List (**Fig 5**). One or multiple electrodes can be selected for further analysis.

- To include channels to process, select them from the *Channels* list and click on the *Add* button.
- To remove a selected channel, click on the electrode in the *Selected Channels* table and click on the *Remove* button.
- The order in which the electrodes are displayed on the *Selected Channels* table is the order in which the electrodes are going to be visualized or processed. If you want to change this order, then click on the respective electrode and click on the *Up* or the *Down* buttons.

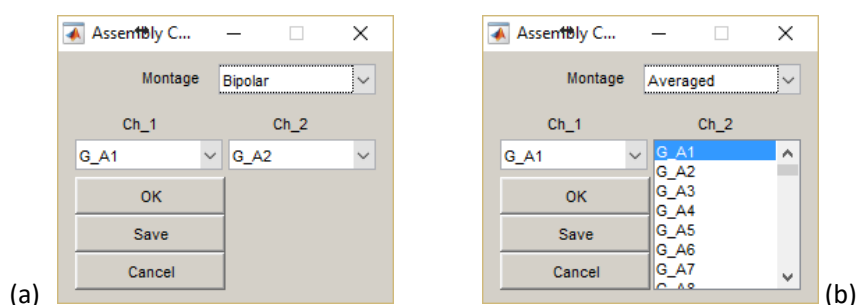
Figure 5. Channel selection in the Select Channels window

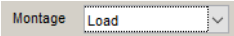


4.2.1. Montage selection

Electrode montages can be included in RIPPLELAB's channel list. For this, click on the *+Montage* button in the *Select Channels* window (**Fig 5**) and the *Assembly Channels* Window will be displayed (**Fig 6**).

Figure 6. Assembly Channels window options




- In the *Assembly Channels* Window you can select the Bipolar (**Fig 6.a**) or average (**Fig 6.b**) montages on the *Montage* popup list. Whereas the *Bipolar* option only gives the possibility to select two channels, the *Average* option give the chance to select multiples electrodes to build the average reference.
- An additional option on the *Montage* popup list is to load a previous built montage selecting the *Load* option . This will open the file explorer to search for a *.rmtg file with the montage configuration.
- Click on the *OK* button to include the new montage on the *Channels* List.

- Click on the *Save* button to save the new montage. This will open the file explorer to save a *.rmtg file with the montage configuration.
- Click on the *Cancel* button to return to the previous window ignoring any selection.

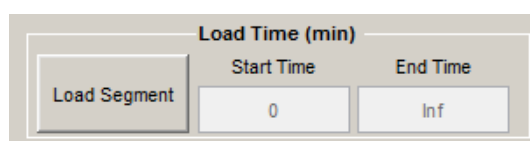
4.3. Channel's Load

Selected electrodes can be further processed by displaying the signals for pre-analysis or by processing the HFO detection without previous visualization.

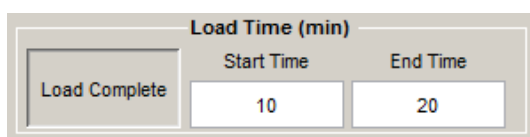
- To display the signals, click on the *Display Ch* button in the *Select Channels* window (**Fig 5**).
- To save Channel information and load the channels only when HFO analysis is performed, click on the *Apply* button in the *Select Channels* window (**Fig 5**).

 It is important to note that RIPPLELAB is set to load all the selected data into memory to further processing. Hence, with working with large files is recommended to select only specific time segments of the focus channels to analyze.

- To load only a signal segment, click on the *Load Segment* button to enable the segment controls of the *Load Time* Box. You can select the start and the end time of the signals to be loaded. Please take in account that start and end time are set in minutes.

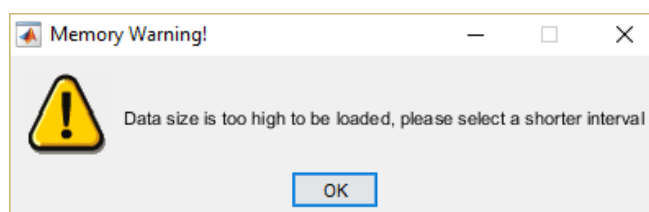


- If a segment is selected and you want to load the complete signal, then click on the *Load Complete* Button to set the start and end times to the first and last sample of the data.

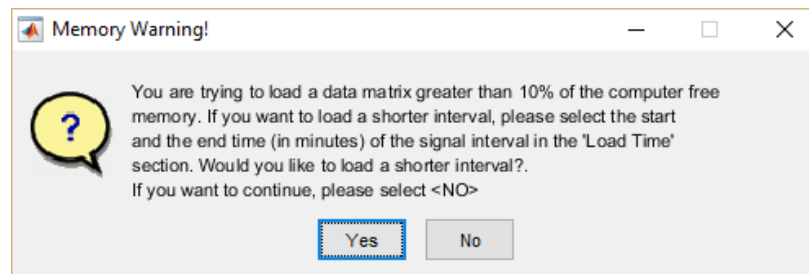


The data size which RIPPLELAB is capable to load depends on the free memory of the system.

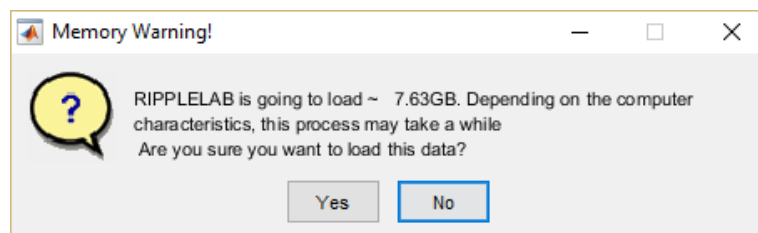
- If data size is larger than 80% of the free memory available, then RIPPLELAB aborts the load and displays a warning message:




- If data size is larger than 10% of the free memory available, then RIPPLELAB displays a warning message indicating this fact and suggest to reduce the data length. To select a shorter data segment click <Yes> in this Message; if you want to continue, then select <No>



- If you have selected to continue the load, a warning message indicating the data size to load is presented. You can abort the load by selecting <No> if the data is too big for the required process.



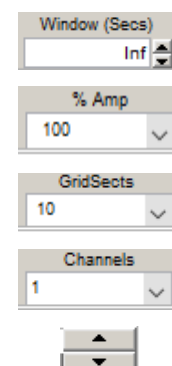
 The time required to load the selected channels selected varies depending on the characteristics of the EEG format to load, the RAM memory of the computer and the number of channels and samples of the data to load. This time fluctuates between some milliseconds and some minutes if data is too large.

4.4. Data pre-processing

If data have been selected to be displayed, the RIPPLELAB's main window (**Fig 2**) allows you to visually explore the raw data and to compute different measures to better tune the HFOs detection algorithms.

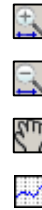
- Visualization controls include [**Fig 2: Visualization controls Box**]:

- Axis management: time window length
- Amplitude scale.
- The number of grids per window.
- Determining the number of channels to be visualized.
- Up/Down Selection



- Zoom and Grid [**Fig 2: Zoom and Grid Options Box**]:

- Horizontal Zoom in
- Horizontal Zoom out
- Panning
- Grid On/Off



- Alternation between relative time (reference to the initial time of the recording) and absolute time (real date and time if present) of the displayed segment of data. [**Fig 2: Window's Time Position Box**]:

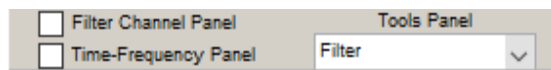
00:00:00.00

- Fast time navigation through the horizontal slider bar

To enable the Filter, Time-Frequency and PSD Panels, the Filter and Time-Frequency analysis must be performed. This is done using the *Panel Display Controls* [**Fig 2: Panel Display Controls box**]. In the *Tools Panel* popup list the option Filter, Time-Frequency and Cursor-Segment Analysis enables the specific options for each of these analyses.

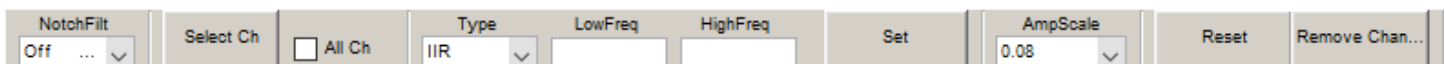
4.4.1. Filtering

The *Filter* option enables the Filter Tools (**Fig 7**).



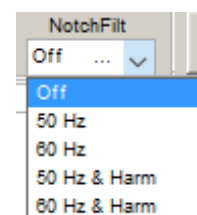
i RIPPLELAB executes the filter analysis over all the signal time of the selected channels. Hence, if the data is large, the time required to complete the filter analysis will take some time.

Figure 7. Filter Tools




- NotchFilter Button:** This popup list selects a notch filter, which applies an fft/iff filter removing the selected frequencies with a smoothed decay in the frequency domain. Specifically, this filter is only computed on all the channels in memory.

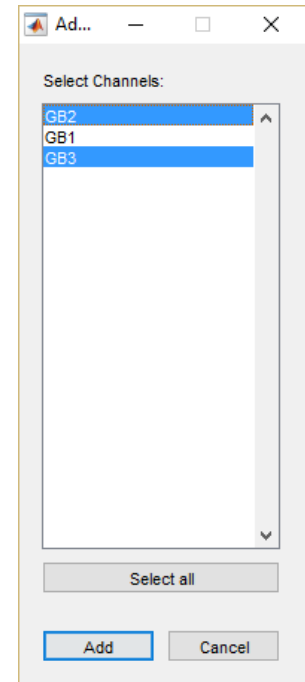
i The filtered data will replace the original data in the main axes.




- **SelectCh Button:** This button opens a list to select the specific channels to filter with custom frequencies.

 The filtered data of the channels selected in this window will be visualized in a parallel panel to the original signal. [Fig 2: Filter Panel]:

 The filtered data will not replace the original data in the main axes.




- **All Ch Checkbox:** When this option is checked, the filter specifications are applied on all the channels in memory.

 The filtered data will replace the original data in the main axes.

- **Type:** This popup list selects the filter type to be applied: IIR (*non-causal filter*) or FIR (*causal filter*). The *causal filter* implements a Hamming-windowed FIR digital filter of 50th order and a cutoff attenuation specified at -6dB (Widmann et al. 2014). The *non-causal filter* employs a type-II Chebyshev IIR forward-backward digital filter, which has a passband ripple of no more than 1 dB and a stopband attenuation of at least 20dB, a cutoff attenuation specified at -3dB, and a second-order section implementation to maintain stability.

- **LowFreq and HighFreq:** Frequency cutoffs of the filter to be implemented. According the values, a filter category is implemented:

- Low Pass Filter: LowFreq = Blank & HighFreq = Numeric value.
- High Pass Filter: LowFreq = Numeric value & HighFreq = Blank.
- Band Pass Filter: LowFreq = Numeric value & HighFreq = Numeric value & LowFreq < HighFreq

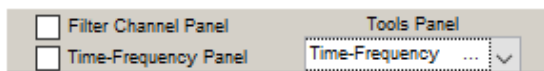
 Non-valid options:

- LowFreq > HighFreq
- LowFreq = Blank & LowFreq = Blank
- Any numeric value > Nyquist Frequency (Sampling frequency / 2)

- **Set Button:** Applies the specific filter depending on the selected parameters
- **AmpScale Popoup List:** When the filter panel is visualized, this option controls the amplitude of the filtered signals.
- **Reset Button:** When the original data has been replaced by the filtered data. This option returns to the original signals.
- **Remove Channels Button:** This button displays a list to select the channels to be removed from the filter panel.

4.4.2. Time Frequency analysis

The *Time-Frequency* option enables the Time-Frequency Tools (**Fig 8**).




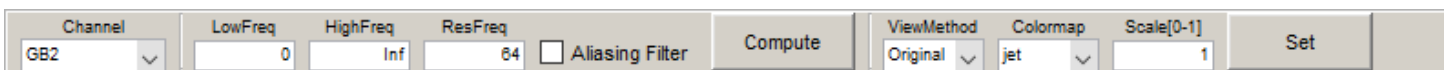

 RIPPLELAB executes the filter analysis only on the displayed segment of the signal. Hence, if the displayed signal is large, the time required to complete the filter analysis will take some time. For each time the displayed segment is computed, the Time-Frequency analysis is computed

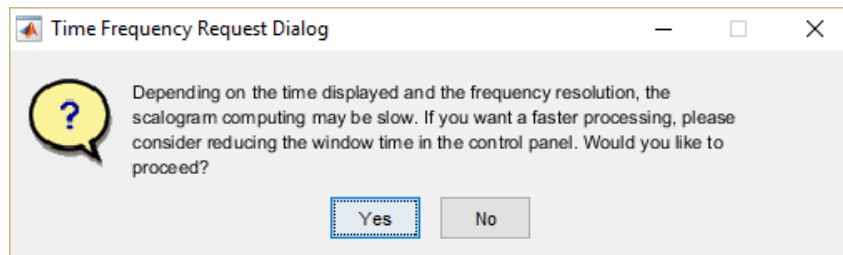
Figure 8. Time-Frequency Tools



- **Channel list:** This control allows you to select the channel to compute the Time-Frequency analysis.
- **LowFreq and HighFreq:** Frequency boundaries for the Time-Frequency analysis. Frequencies must be $\text{LowFreq} < \text{HighFreq} < \text{Nyquist Frequency (Sampling frequency / 2)}$
- **ResFreq:** This control allows you to select the Frequency resolution. The numeric value corresponds to the number of frequency steps between the low frequency boundary and the high frequency boundary.
- **Aliasing Filter checkbox:** When this option is checked, an antialiasing filter is computed before the Time-Frequency analysis.
- **Compute button:** Computes the Time-Frequency analysis.

 The Time-Frequency analysis of the selected channel will be visualized in a parallel panel to the original signal.
[Fig 2: Time-Frequency Panel]:

 A warning message will be displayed:



- **ViewMethod list:** When the Time-Frequency panel is displayed, this option allows you to change the equalization of the Time-Frequency coefficients.
- **Colormap list:** When the Time-Frequency panel is displayed, this option allows you to change the colormap of the visualized Time-Frequency.
- **Scale:** This control requires a numeric value in the interval $[0,1]$ which scales the Time-Frequency coefficients amplitude.
- **Set button:** Clicking in this button applies any changes established in the *ViewMethod*, *Colormap* and *Scale* controls.

4.4.3. Cursors measures

The *Cursor-Segment Analysis* option enables the Cursor and Spectrum Tools (**Fig 9**).

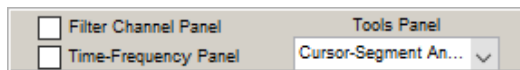


Figure 9. Cursor and Spectrum Tools



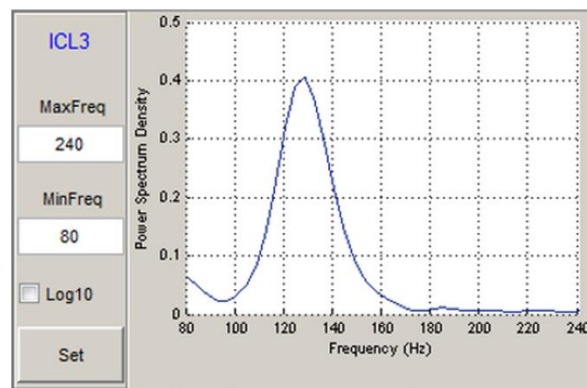
- **Channel list:** This control allows you to select the channel to compute cursor measures and power spectrum.
- **Cursor_1 and Cursor_2 buttons:** These buttons enables and disables the cursors 1 and 2.
- **Cursor Measures:** These controls display different measures depending on the cursors position.
 - **Cur1_x, Cur_2x, Cur1_y, Cur_2y:** Display the cursors positions in the x and y axes.
 - **TimeInt:** Time interval between cursors in seconds.
 - **AvgVal:** Average amplitude value of the signal segment between cursors.
 - **MinVal, MaxVal:** Minimum and Maximum amplitude values of the signal segment between cursors.

- **Power Spectrum button:** Computes the power spectrum of the signal segment between cursors, and it displays the analysis on the PSD Panel [Fig 2: PSD Panel].

4.4.4. Power spectrum

The PSD panel (Fig 10) allows the user to set basic display options of the power spectrum graph.


Figure 10. The PSD Panel



- **MinFreq and MaxFreq:** Frequency boundaries for the power spectrum analysis. Frequencies must be $\text{MinFreq} < \text{MaxFreq} < \text{Nyquist Frequency (Sampling frequency / 2)}$.
- **Log10 checkbox:** Visualize Logarithmic or linear Frequency axis.
- **Set Button:** Clicking in this button applies any changes established in the *MinFreq*, *MaxFreq* and *Log10* controls
- Move the cursors by grabbing the triangles on the cursor bars



5. HFOs DETECTION

To detect HFOs, click on the *HFO Detection Methods* icon  [Fig 2: HFO Analysis Tool Box], or you can go the menu *Tools>HFO Detection Methods*. This will open the *HFO-Detection Methods* window (Fig 11).

To process an automatic HFO detection, it is not necessary that the signals to analyze had been previously displayed. If you click on the *Apply* button in the *Select Channels* window (Fig 3), the signals are going to be loaded only when required by the HFO detection process

5.1. General Controls

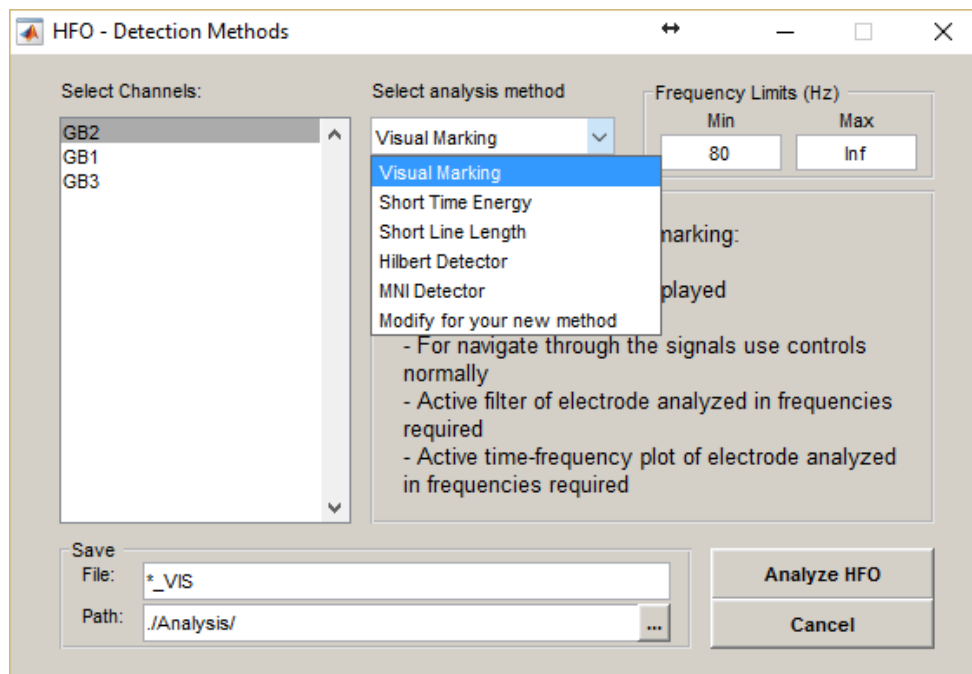
- **Select Channels list:** This control allows you to select the channels to analyze using the HFO detection methods



Multiple electrodes can be selected for automatic detection of HFOs, but only one electrode is possible when selecting the visual marking analysis. [Fig 2: Time-Frequency Panel]:

- **Select analysis method list:** This control allows you to select the HFO detection method.
- **Frequency Limits:** This control allows you to select frequency interval in which the HFO detection is developed. Frequencies must be $\text{Min} < \text{Max} < \text{Nyquist Frequency (Sampling frequency / 2)}$.
- **Save Controls:** These controls allow you to select the Name and the Path for the results file for the HFO analysis. The sign '*' acts as a wildcard character which matches the original file name.
- **Analyze HFO:** This button starts the HFO detection process.
- **Cancel Button:** This option ignores all the parameters and returns to the previous window.

Figure 11. HFO-Detection Methods window



5.2. Manual detection of HFOs

- To perform a manual detection of HFOs, please select the *Visual Marking* option on the Select Analysis method list (Fig 11).



Manual detection can only be performed for a single electrode, which must be first displayed according to the procedure described in the 4.2 and 4.3 sections.

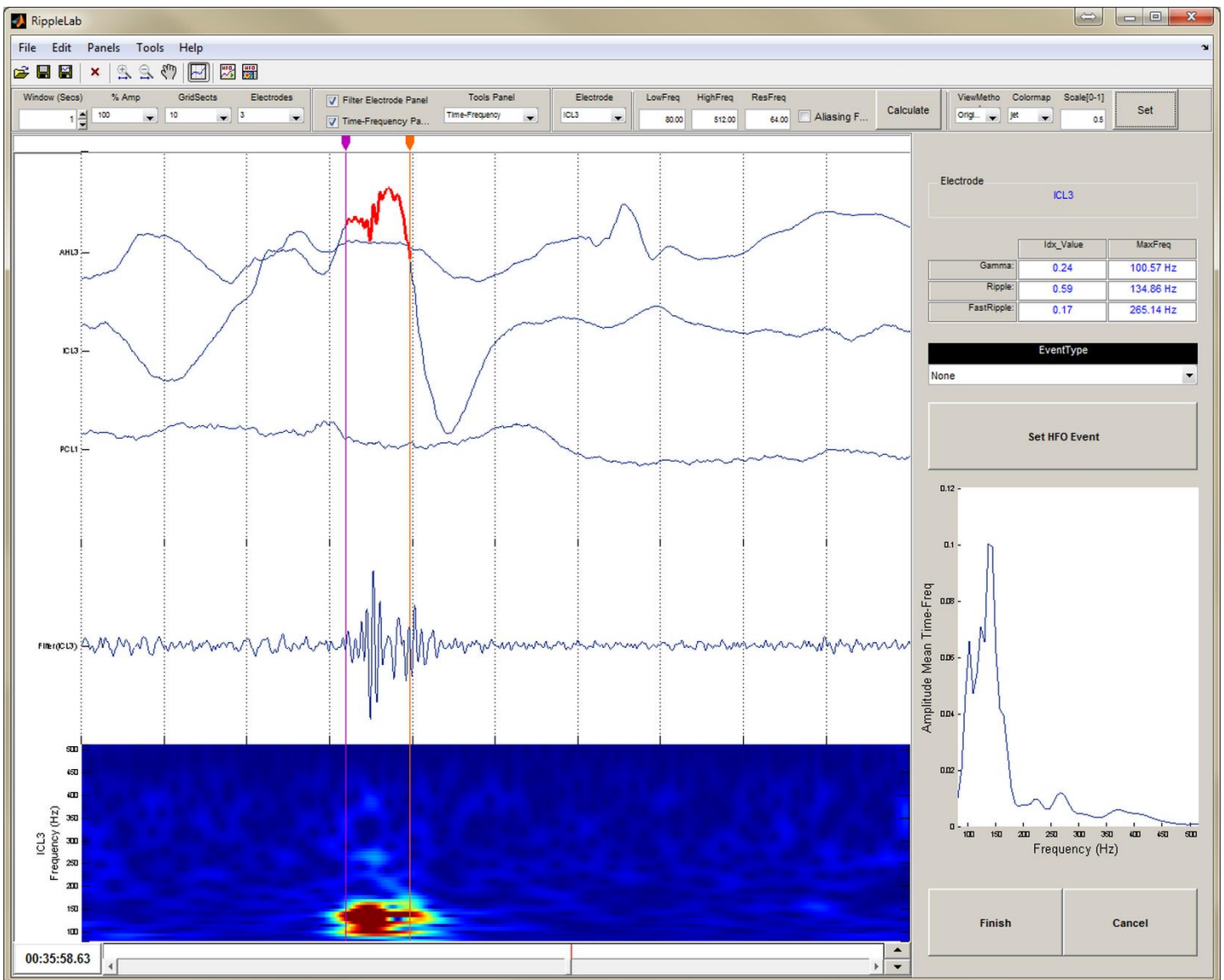
- b. In the *Frequency Limits* panel, select the frequency range where the visual detection will be performed.
- c. Select the name and path of the results file in the *Save* panel.
- d. Click on the Analyze HFO button. Then, the main figure is set to manually mark HFOs as displayed in **Fig 12**.



As the Time-Frequency analysis is developed in the selected frequency range, then the warning message of 4.4.2 section is displayed.

- e. If you need, you can add as many filtered plots in the desired frequencies as described in section 4.4.1.
- f. Look for HFOs. When a HFO is found, set the event limits with the cursors and click on the *Set HFO Event* button.

Figure 12. RIPPLELAB's main figure set to develop the HFO visual marking



- HFO visual marking controls include (**Fig 12**)

- Channel/Electrode: Label of the electrode to be analyzed
- The gamma, ripple and fast ripple indexes (Ibarz et al. 2010)
- Set Event Type for detailed classification of the selected event as Gamma, Ripple or Fast Ripple.
- Set HFO Button, when a HFO is found
- Use the power spectrum density to have more frequency information about the signal between cursors
- Click on the finish button to save the HFO marks, or click on the cancel button to exit the visual marking mode

	Idx_Value	MaxFreq
Gamma:	0.24	100.57 Hz
Ripple:	0.59	134.86 Hz
FastRipple:	0.17	265.14 Hz

5.3. Automatic detection of HFOs

Four automatic methods for HFO detection were implemented in RIPPLELAB: the Short Time Energy detector (Staba et al. 2002), the Short line length detector (Gardner et al. 2007), the Hilbert detector (Crépon et al. 2010) and the MNI detector (Zelmann et al. 2012). Each of these methods have their own specific parameters (**Fig 13**) and default values are established as published by each work. A brief description of each method is stated at next.

- The **Short Time Energy (STE)** is the algorithm proposed by Staba et al., 2002. Setting parameters are displayed in **Fig 13.b**. In brief, the wideband EEG signal is band-pass filtered in the high frequency range (*Frequency Limits* parameter). The energy from the filtered signal is then computed using the root mean square (RMS) defined by the equation

$$E(t) = \sqrt{\frac{1}{N} \sum_{k=\ell-N+1}^{\ell} x^2(k)}$$

Figure 13. Automatic detection parameters

a)

b)

c)

d)

e)

within a $N = 3$ -ms window (*RMS_Window* parameter), and successive RMS values greater than 5 standard deviations (SD) above the overall RMS mean (*RMS SD_Threshold* parameter) are selected as putative HFO events if they last more than 6 ms (*Min_Window* parameter). Finally, only events containing more than 6 peaks (*Min Oscillations* parameter) greater than 3 SD above the mean value of the rectified band-pass signal are retained (*Peak SD_Threshold* parameter). In addition, events separated by 10 ms or less are marked as a single oscillation (*Min Gap Time* parameter). In the original paper, the energy threshold was established to be set depending to the complete analyzed segment, which had a duration of 10-min. Hence in RIPPLELAB, the energy threshold can be computed for the entire signal, as originally proposed by Staba, or shorter segments can also be used, as suggested by Gardner et al. (2007) (*Epoch* parameter).

- The **Short Line Length detector (SLL)** was developed by Gardner et al. (2007) and Worrell et al. (2008); setting parameters are displayed in **Fig 13.c**. In this approach, a preprocessing stage is done with a derivative filter in order to equalize the spectrum of the signal. Next, a band-pass filter is applied (*Frequency Limits* parameter). Then, the energy of the signal is calculated by a short time line length measure (Esteller et al. 2001) defined by

$$E(t) = \sum_{k=t-N+2}^i |x(k) - x(k-1)|$$

with window $N = 5$ ms (*Filter Window* parameter). An event is valid if its amplitude is greater than the 97.5th percentile of the empirical cumulative distribution function (*Threshold Percentile* parameter) for each 3-min epoch (*Epoch* parameter) and if it has a minimum duration (*Event Min Time* parameter). This duration is set in 80 ms in Gardner et al. (2007), but it is ignored in Worrell et al. (2008). We set this parameter as 12 ms by default in order to accept events larger than 6 oscillations at 500Hz.

- The **Hilbert Detector (HIL)** was proposed by Crépon et al., (2010). Setting parameters shown in **Fig 13.d**. In this method, the signal is first filtered between a selected frequency range (*Frequency Limits* parameter), and the envelope is then

computed with the Hilbert transform. For an event to be considered valid, two conditions must be met: first, for each event, the local maximum must exceed a threshold of 5 SD of the envelope (*SD Threshold* parameter) calculated originally over the entire recording or from a time interval (*Epoch* parameter). Second, each detected HFO must have a minimal time length of 10 ms (*Min EventTime* parameter). As in the STE detector case, we included the possibility to analyze the threshold by epochs specified by the user.

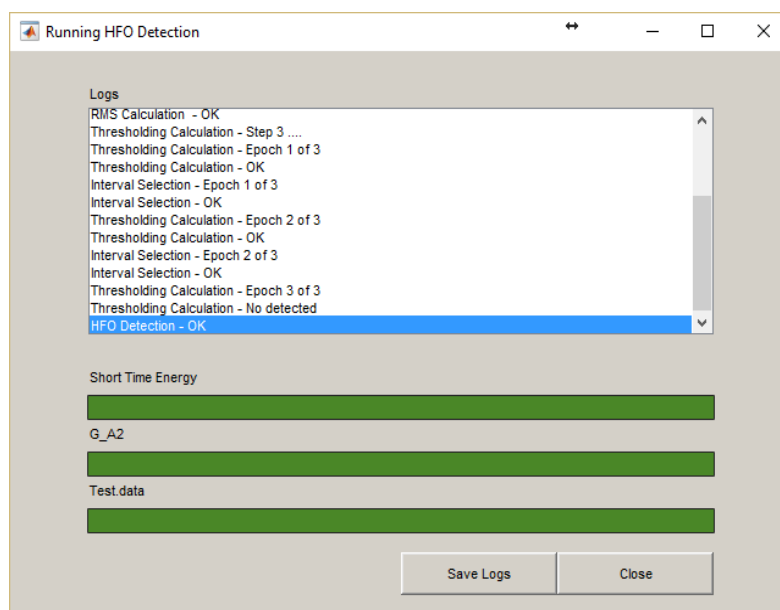
- **The MNI detector (MNI)** was developed by Zelmann et al. (2012). Setting parameters are observed in **Fig 13.e**. In this method the signal is first band-pass filtered (*Frequency Limits* parameter). Then, a baseline detection procedure based on the wavelet entropy is applied (Zelmann et al. 2010). For this, the signal is divided into segments of 125 ms (*Baseline Segment* parameter) with 50% overlap (*Baseline Shift* parameter). Next, for each segment, the normalized wavelet power of the autocorrelation function is computed using the complex Morlet wavelet (Morlet et al. 1982). Subsequently, the maximum theoretical wavelet entropy from the segment is obtained for the white noise (Rosso et al. 2001), and the segment is considered as a baseline interval when the minimum entropy is larger than a threshold (*Baseline Threshold* parameter). If a “sufficient amount” of baseline exists, HFOs candidates are detected in accordance with the energy, defined as the moving average of the RMS amplitude of the filtered signal. Segments with energy above a threshold (*Threshold Percentile* parameter) and lasting more than 10 ms (*MinWindow* selection) are considered as HFO. Similar to other methods, events located less than 10 ms apart are considered as single events (*Min GapTime* option). If a sufficient amount of baseline is not present in the signal, an iterative procedure is carried out where the threshold is computed for the band-passed signal. Originally, this detection methodology was implemented with 1-min segments of EEG signal. Therefore, we included the possibility to process the data thresholds in epochs of time specified by the user.


5.4. HFO detection process and logs

To start an automatic HFO detection process, click on the *Analyze HFO* button in the *HFO-Detection Methods* window (**Fig 11**). This selection launch the *Running HFO Detection* window (**Fig 14**), which indicates the progress of the HFOs detection process.

After the detection process ends, you can save the detection logs in a text file by clicking on the **Save Logs** button. Click on the Close button to return to the main window

Figure 14. Running HFO detection window



 After any detection process is finished and the signals have been previously displayed, a group of red marks will appear upon the horizontal slider bar [Fig 2. Long Term Detected HFO Box]. Each mark is a shortcut to each detected event, so if you click on these marks, the time position skips to the selected HFO event.

6. HFOs VALIDATION


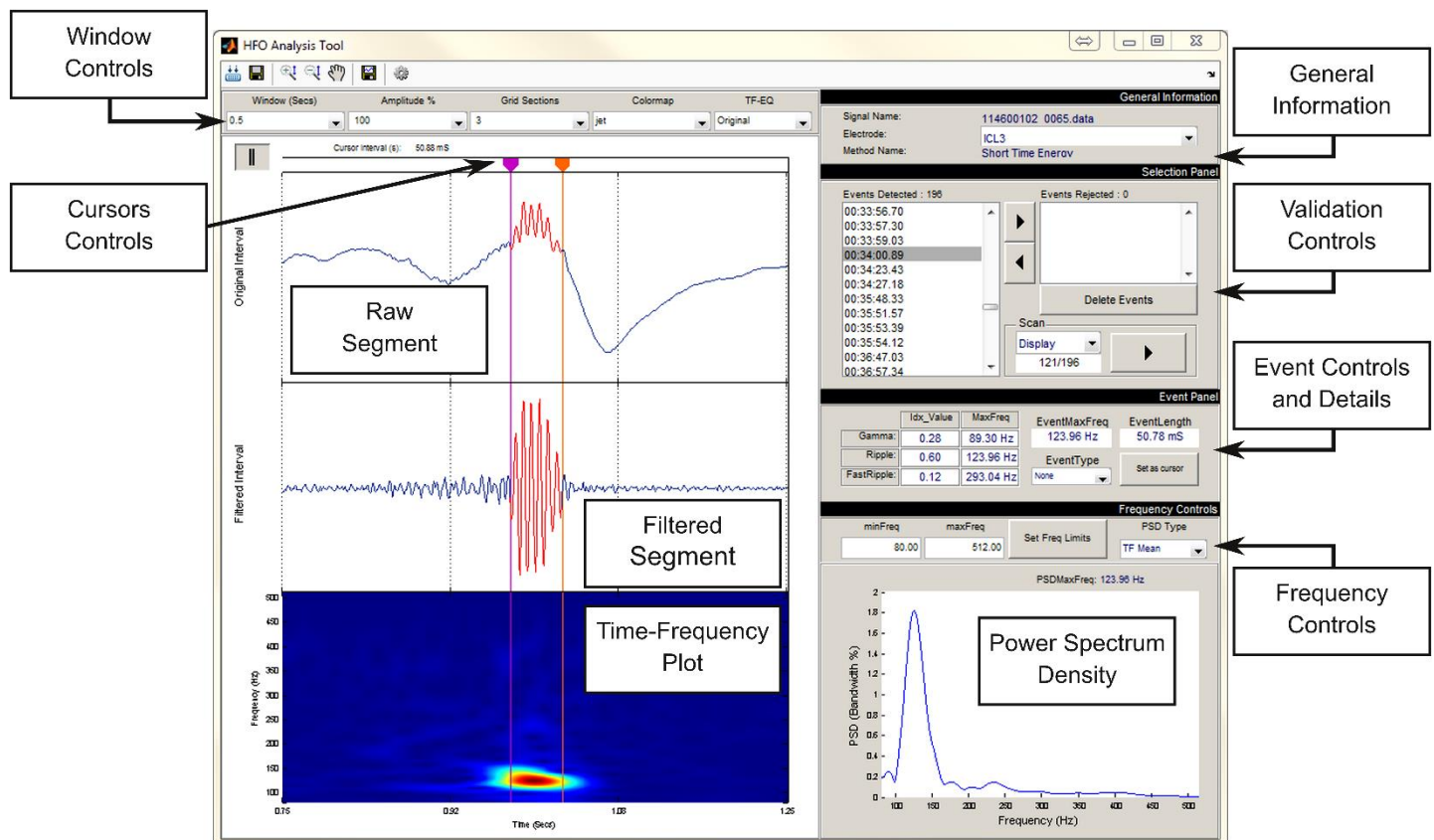
To open the tool for HFO validation, click on the *HFO Analysis Tool* icon  [Fig 2: HFO Analysis Tool Box], or you can go the menu Tools>HFO Analysis Tool. This will open the HFO Analysis Tool interface (Fig 15).

Figure 15. HFO Analysis Tool interface



- Window controls include [Fig 15: Window controls Box]:
 - Window (Sec):** time window length
 - Amplitude** scale in %.
 - Grid Sections** changes the number of grids per window.
 - Colormap** changes the Time-Frequency scale of colors

Window (Secs)

0.5

Amplitude %

100

Grid Sections

5

Colormap

jet

- **TF/EQ** changes the equalization of the Time-Frequency coefficients
- General Information Panel [Fig 15: General Information Box]:
 - It presents the name of the analyzed file, the selected method for detection and the currently selected channel
- Selection Panel [Fig 15: Validation Controls Box]:
 - **Events detected** displays the time moments where events were detected. By clicking each row, the corresponding event is selected and displayed.
 - **Events Rejected** displays the time moments where events were manually rejected.
 - **Accept/Rejected controls** shift the events as rejected or accepted depending on the selection.
 - **Delete Events button** clears all the rejected events
 - **Scan** offers three different possibilities.
 - *Display* allows you to automatically visualize all detected events.
 - *Auto-Classify* lets an event classification based on its spectral characteristics. Thus, they can be labeled as *Gamma*, *Ripple*, *Fast Ripple*, *Spikes*, *Artifacts* or *Others*.
 - *Remove Others* allows you to delete those events classified as *Others*.
- Event Panel [Fig 15: Event Controls and Details Box]:
 - The gamma, ripple and fast ripple indexes (Ibarz et al. 2010)
 - **EventMaxFreq** Maximum frequency of the signal for the segment between cursors.
 - **EventLength** Time in seconds of the selected event.
 - **EventType** allows you define the event type according to the classification: *Gamma*, *Ripple*, *Fast Ripple*, *Spike*, *Artifact* or *Other*.

TF-EQ

Original

General Information

Signal Name: 114600102_0065.data

Electrode: ICL3

Method Name: Short Time Energy

Events Detected : 196

00:35:51.57

00:35:53.39

Events Rejected : 0



Delete Events

Scan

Display

130/196

	Idx_Value	MaxFreq
Gamma:	0.27	83.44 Hz
Ripple:	0.73	127.91 Hz
FastRipple:	0.00	Hz

EventMaxFreq

127.91 Hz

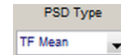
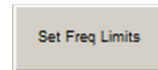
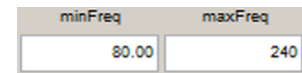
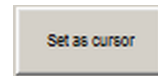
EventLength

45.90 mS


EventType

None

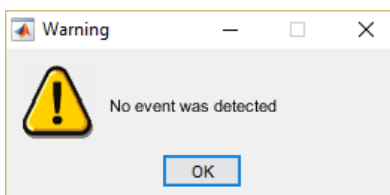
- **Set as a cursor** Time in seconds of the selected event.
- Frequency Controls Panel [Fig 15: Validation Controls Box]:
 - **minFreq/maxFreq** allows the user to change the frequency limits for both the filtered signal and the time-frequency plot.
 - **Set Freq Lims button** establishes any change in the frequency limits
 - **PSD Type** determines the computation type of the power spectrum for the segment between cursors.
- The left panel presents a short segment of individual events.
 - **Clip to event button** sets automatically the cursors to the detected event segment.
 - In the top panel, the raw signal is plotted highlighting the HFO in red. The equivalent filtered signal and the time-frequency plot are presented in middle and bottom panels, respectively.



6.1. Load a HFO analysis


After starting the HFO Analysis Tool Interface, click on the Import *Analysis Icon*  to load a HFO analysis file (*.rhfe).

If the HFO analysis file contains detected events, they will be automatically presented. Conversely, if no events were detected, a warning message will be displayed




6.2. Review a HFO analysis

- a. You can visually validate each of the detected events by selecting the different events in the *Events Detected* list. Move throughout the events by clicking on them or with the arrow keys.
- b. Reject a selected event by clicking on the remove button (Right Arrow Button) [Fig 15: Validation controls Box].
- c. Delete the rejected events by clicking the *Delete Events* button [Fig 15: Validation controls Box].
- d. Accept a rejected event by selecting it in the *Events Rejected* list and clicking on the accept button (Left Arrow Button) [Fig 15: Validation controls Box]
- e. Classify the event type as *Gamma*, *Ripple*, *Fast Ripple*, *Spike*, *Artifact* or *Other* using the *EventType* control [Fig 15: Event Controls and Details Box]

 To speed up the validation process, some keyboard shortcuts have been implemented:

- Shift + arrow left or Shift + arrow right: controls the position of the left cursor.
- Ctrl + arrow left or Ctrl + arrow right: controls the position of the right cursor.
- Alt + Enter: event limits are resized to the position limited by the cursors.
- D key: moves the selected event to the Events Rejected list.
- A key: moves the last event in the Events Rejected list to the Events Detected list.

6.3. Save a HFO analysis

After the review process, save the HFO analysis file by clicking on the save analysis button .

7. RIPPLELAB SCRIPTS

This section is related to specific characteristics of RIPPLELAB programming. This information is not relevant for the usage of the RIPPLELAB interface, but it gives useful material for the understanding of RIPPLELAB scripts and organization.

7.1. Scripts overview

RIPPLELAB GUI is completely written programmatically without the use of MATLAB GUIDE. Consequently, all the figures included in the RIPPLELAB interface are not previously created as .fig objects.

RIPPLELAB's programming establishes a variable prefix code which indicates the variable type as described at next:

- *s_VariableName*: The (s_) prefix states the variable use as a scalar (*single numeric value*).
- *v_VariableName*: The (v_) prefix states the variable use as a vector (*mx1 or 1xn array*).
- *m_VariableName*: The (m_) prefix states the variable use as a matrix (*mxnrx... array*).
- *st_VariableName*: The (st_) prefix states the variable use as a structure or MATLAB object.
- *str_VariableName*: The (str_) prefix states the variable use as a character string.
- *f_FunctionName*: The (f_) prefix states the name of a function or nested function.
- *p_ProgramName*: The (p_) prefix states the name of a function used as the main script of a code (program).

The scripts are written in MATLAB sections named according to the functionality of the code, and the object handles are stored in MATLAB structures.

To facilitate the process of edition, the sections to modify when a new method is included are marked with the comment: `[**INSERT!**]`, and an example is provided for each case.

7.2. Including a new EEG format

To include a new EEG format load script, two features functions must be modified: `f_GetHeader()` for EEG header reading and `f_GetData()` for EEG samples data reading.

- For `f_GetHeader` find the `[**INSERT!**]` comment and follow the example. A new case section referenced to the file extension must be included. For Instance, for a new EEG format with file extension `*.new` and header loader script named `"newformatreadheader()"`, the case section may be established as:

```
function st_Info = f_GetHeader(pst_SigPath)
% Reads header information from header or eeg files
% INPUT:
% pst_SigPath.name: Name of Signal file
% pst_SigPath.path: Path of Signal File
% OUTPUT
% st_Info.SigPath      = Full Path of signal file
% st_Info.SigName      = Signal Name;
% st_Info.SigExt       = File extension for signal type;
% st_Info.str_FileType = File Type
% st_Info.Start        = Absolute start time
% st_Info.Time         = Time length of record in mins;
% st_Info.SampleRate   = Sample rate for signal;
% st_Info.NumbRec      = Number of records;
% st_Info.Labels       = Label of records;
% st_Info.Scale        = Amplitude scale;
% st_Info.error        = Reading Error;
% st_Info.AmpScaleRec  = Amplitude scale for record;
% st_Info.MinMaxRec    = min value and max value per record;
% st_Info.Custom       = Structure with custom information according with type

% Script
(...)

case 'new'
    st_Info = newformatreadheader(str_FullPath);

    st_Info.s_Start      = st_Info.a;
    st_Info.s_Time       = st_Info.b;
    st_Info.s_Samples    = st_Info.c;
    st_Info.v_SampleRate = st_Info.d;
    st_Info.s_NumbRec    = st_Info.e;
    st_Info.v_Labels     = st_Info.f;
    st_Info.s_Scale      = st_Info.g;
    st_Info.st_Custom    = st_Info;

% Script
(...)
```

- For `f_GetData` find the `[**INSERT!**]` comment and follow the example. A new case section referenced to the file extension must be included. For Instance, for a new EEG format with file extension `*.new` and data loader script named `"newformatreaddata ()"`, the case section may be established as:

```
function st_Data = f_GetData(pst_Info,pv_TimeLims,pv_Selected)
% Reads channels data from eeg files
% INPUT:
% pst_Info = Information structure
%
% st_Info.SigPath = Full Path of signal file
% st_Info.SigName = Signal Name;
% st_Info.SigExt = File extension for signal type;
% st_Info.str_FileType= File Type
% st_Info.Start = Absolute start time
% st_Info.Time = Time length of record in mins;
% st_Info.SampleRate = Sample rate for signal;
% st_Info.NumbRec = Number of records;
% st_Info.Labels = Label of records;
% st_Info.Scale = Amplitude scale;
% st_Info.error = Reading Error;
% st_Info.AmpScaleRec = Amplitude scale for record;
% st_Info.MinMaxRec = min value and max value per record;
% st_Info.Custom = Structure with custom information according with
% type
%
% pv_TimeLims = Time limits of channels to load
% pv_Selected

% OUTPUT
%
% st_Data = Data structure
%
% st_Data.v_Labels = Labels of selected channels;
% st_Data.s_Sampling = Frequency sample;
% st_Data.v_TimeLims = Time limits of channels to load;
% st_Data.m_Data = mxn matrix of channels samples m: samples, n:channels
% st_Data.v_Time = Time vector;
% st_Data.s_TotalTime = Total time in seconds;

% Script
(...)

case 'new'

% f_AskSamplesLims() gives the sample position of the selected start and end
% time

[s_Start,s_End] = f_AskSamplesLims(st_Data.s_Sampling,...
    pst_Info.s_Time,pv_TimeLims);

if isempty(s_End)
    s_Start = ... % Stablish here the start condition for reading the first
    sample
end

if isempty(s_End)
    s_End = ... % Stablish here the start condition for reading the last
    sample
end

st_Data.m_Data = newformatreaddata(pst_Info.str_SigPath,...
    'channels',pv_Selected,...
```

```

        'start',s_Start,...
        'stop',s_End);

    if find(size(st_Data.m_Data) == numel(pv_Selected)) == 1
        st_Data.m_Data = st_Data.m_Data';
    end

% Script
(...)

```

7.3. Including a new detection method

To include a new detection method, the following two features must be set: the visualization of the panel that allows the configuration of different parameters associated with the new method and the selection of the new method for further processing.

In order to access and visualize the new method settings, the cell array `v_HFOMethodsList` containing the HFO detection options must be modified as follows:

```

v_HFOMethodsList = [{'Visual Marking'}, {'Short Time Energy'},...
                    {'Short Line Length'}, {'Hilbert Detector'},...
                    {'MNI Detector'}, {'Modify for your new method'}];

% Example:
% v_HFOMethodsList = [{'Visual Marking'}, {'Short Time Energy'}, ...
%                    {'Short Line Length'}, {'Hilbert Detector'},...
%                    {'MNI Detector'}, {'XXX Detector'}];

```

If some new selection controls for the method to implement are desired, a new panel must be enabled in the section “%% - [Controls] << Auxiliary Figures >> - (HFO Detection Methods)”, and the handle objects corresponding to HFO settings should be saved in the structure named `st_HFOMethod` (e.g. `st_HFOMethod.XXX_Epoch`, `st_HFOMethod.XXX_Threshold`, `st_HFOMethod.XXX_Window`, etc). The corresponding parameter controls should be defined in the section identified as “%% - [Controls] || HFO DETECTION - PANEL || New Method Objects ||”, and the new panel visibility must be set in the `f_HFOSelectMethod()` nested function under the section “%% [Functions] Auxiliary Figure - Select and Run HFO Methods”.

To set the selection of the method for processing, the user must include the new detection case in the switch block defined inside the `f_HFORunMethod()` function. This block calls the nested functions to set the algorithm parameters of each method. A new nested function must be created, and it can be named as desired (e.g. `f_HFOSet-XXX`). The goal of this new nested function is to establish all the parameters required specifically for the new implemented algorithm and to call the external function that performs the detection method. This nested function is placed in the section “%% [Sub-Functions] HFO Methods: New Method” and an example is stated as follows:

```

function f_HFOSet-XXX()
%Function to read the settings for the new implemented method XXX
st_HFOSettings.s_Epoch      = eval(get(st_HFOMethod.XXX_Epoch,'string'))
st_HFOSettings.s_Thresh     = eval(get(st_HFOMethod.XXX_Threshold,'string'));

```

```

st_HFOSettings.s_Window      = eval(get(st_HFOMethod.XXX_Window, 'string'));
st_HFOAnalysis.str_DetMethod = 'Method XXX';

st_HFOAnalysis.m_EvtLims = f_findHFOxxxx(...
    str_TempFile, ...
    st_HFOControl.s_CurrElecIdx, ...
    st_HFOSettings, ...
    st_HFOAnalysis.s_Sampling);

end

```

In the previous example, the structure st_HFOSettings contains the specific settings for the XXX method, which also includes the frequency band for the analysis, and f_findHFOxxxx() corresponds to the external function where the detection algorithm is implemented. The values for str_TempFile, st_HFOControl.s_CurrElecIdx and st_HFOAnalysis.s_Sampling are already set by the system, and they correspond respectively to the path where the data is stored, the index of the channel to be analyzed and the sampling frequency of the data. The f_findHFOxxxx() function must return the detected events in a mx2 matrix where each row is a putative event, the first column corresponds to the first index of the detected event and the second column corresponds to the last index of the detected event. It is important to note that the function output must be called st_HFOAnalysis.m_EvtLims for subsequent processing in RIPPLELAB. An example of the external function implementing the detection algorithm is presented in the pseudocode as follows:

```

function m_HFOEvents = f_findHFOxxxx(pstr_SignalPath, ps_SignalIdx, ...
st_HFOData, s_SampleFrec)

m_Data      = [];
load(pstr_SignalPath)
v_Signal     = m_Data(:,ps_SignalIdx);
clear m_Data

v_Freqs      = [st_HFOData.s_FreqIni st_HFOData.s_FreqEnd]; % Filter freqs
s_Epoch      = st_HFOData.s_Epoch; % Epoch Time
s_Thresh     = st_HFOData.s_Thresh; % Threshold
s_Window     = st_HFOData.s_Window; % Window Time
clear st_HFOData

% Preprocessing Filter
(...)
% Thresholding Process
(...)
% Event detection
(...)

if Events Detected
    m_HFOEvents = [v_IdxIni(:) v_IdxEnd(:)];
else
    m_HFOEvents = [];
end

% v_IdxIni: vector with the initial indexes of the detected events
% v_IdxEnd: vector with the final indexes of the detected events

end

```


8. Appendixes

8.1. GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program—to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the

public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users’ Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work’s users, your or third parties’ legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a

particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

9. REFERENCES

- Crépon, B., Navarro, V., Hasboun, D., Clemenceau, S., Martinerie, J., Baulac, M., et al. (2010). Mapping Interictal Oscillations Greater than 200 Hz Recorded with Intracranial Macroelectrodes in Human Epilepsy. *Brain : A Journal of Neurology* 133 (Pt 1): 33–45. doi:10.1093/brain/awp277.
- Esteller, R., Echauz, J., Tcheng, T., Litt, B., and Pless, B. (2001). Line Length: An Efficient Feature for Seizure Onset Detection. *2001 Conference Proceedings of the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society* 2.
- Gardner, AB., Worrell, GA., Marsh, E., Dlugos, D., and Litt, B. (2007). Human and Automated Detection of High-Frequency Oscillations in Clinical Intracranial EEG Recordings. *Clinical Neurophysiology : Official Journal of the International Federation of Clinical Neurophysiology* 118 (5): 1134–43. doi:10.1016/j.clinph.2006.12.019.
- Ibarz, JM., Foffani, G., Cid, E., Inostroza, M., and Menendez de la Prida, L. (2010). Emergent Dynamics of Fast Ripples in the Epileptic Hippocampus. *The Journal of Neuroscience : The Official Journal of the Society for Neuroscience* 30 (48): 16249–61. doi:10.1523/JNEUROSCI.3357-10.2010.
- Morlet, J., Arensz, G., Fourgeau, E., and Giard, D. (1982). Wave Propagation and Sampling Theory: Sampling Theory and Complex Waves. *Geophysics* 41 (2): 222–36. doi:10.1190/1.1441329.
- Oostenveld, R., Fries, P., Maris, E., and Schoffelen, J-M. (2011). FieldTrip: Open Source Software for Advanced Analysis of MEG, EEG, and Invasive Electrophysiological Data. *Computational Intelligence and Neuroscience* 2011 (January): 156869. doi:10.1155/2011/156869.
- Rosso, O., Blanco, S., Yordanova, J., Kolev, V., Figliola, A., Schürmann, M., et al. (2001). Wavelet Entropy: A New Tool for Analysis of Short Duration Brain Electrical Signals. *Journal of Neuroscience Methods* 105 (1): 65–75. <http://www.ncbi.nlm.nih.gov/pubmed/11166367>.
- Staba, RJ., Wilson, CL., Bragin, A., Fried, I., and Engel, J. (2002). Quantitative Analysis of High-Frequency Oscillations (80-500 Hz) Recorded in Human Epileptic Hippocampus and Entorhinal Cortex. *Journal of Neurophysiology* 88 (4): 1743–52. <http://www.ncbi.nlm.nih.gov/pubmed/12364503>.
- Valderrama, M., and Le Van Quyen, M. (2011). High-Frequency Oscillations and Interictal Spikes in Partial Epilepsy: Joining the Benefits. *Clinical Neurophysiology* 122 (1). International Federation of Clinical Neurophysiology: 3–4. doi:10.1016/j.clinph.2010.06.006.
- Vidaurre, C., Sander, TH., and Schlögl, A. (2011). BioSig: The Free and Open Source Software Library for Biomedical Signal Processing. *Computational Intelligence and Neuroscience* 2011: 935364. doi:10.1155/2011/935364.
- Widmann, A., Schröger, E., and Maess, B. (2014). Digital Filter Design for Electrophysiological Data - a Practical Approach. *Journal of Neuroscience Methods* 250. Elsevier B.V.: 34–46. doi:10.1016/j.jneumeth.2014.08.002.
- Worrell, GA., Gardner, AB., Stead, SM., Hu, S., Goerss, S., Cascino, GJ., et al. (2008). High-Frequency Oscillations in Human Temporal Lobe: Simultaneous Microwire and Clinical Macroelectrode Recordings. *Brain : A Journal of Neurology* 131 (Pt 4): 928–37. doi:10.1093/brain/awn006.
- Zelmann, R., Mari, F., Jacobs, J., Zijlmans, M., Chander, R., and Gotman, J. (2010). Automatic Detector of High Frequency Oscillations for Human Recordings with Macroelectrodes. *Conference Proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference* 2010 (January): 2329–33. doi:10.1109/IEMBS.2010.5627464.
- Zelmann, R., Mari, F., Jacobs, J., Zijlmans, M., Dubeau, F., and Gotman, J. (2012). A Comparison between Detectors of High Frequency Oscillations. *Clinical Neurophysiology* 123 (1): 106–16. doi:10.1016/j.clinph.2011.06.006.