

Counting Sheep PSG



User Guide

Version 1.2

April 2025

Copyright (C) Stuart Fogel & Sleep Well, 2024



User Guide

Counting Sheep PSG



Counting Sheep PSG is EEGLAB-compatible analysis software for manual and visual sleep stage scoring, signal processing and event marking of polysomnographic (PSG) data for MATLAB.

Contents

1. Getting started	-----	1
• Requirements		
• Download and installation		
• Software registration		
2. Display montage settings	-----	3
3. Importing and displaying PSG datasets	-----	5
• Importing PSG data to EEGLAB format		
• Displaying and navigating a dataset		
4. Preprocessing	-----	7
• Downsample data		
• Re-referencing channels		
• Edit channels		
• Filter channels		
5. Artifact correction and signal processing	-----	11
• Mark bad channels		
• Interpolate bad channels		
• Artifact subspace reconstruction (ASR)		
• Independent component analysis (ICA)		
• Power spectral analysis		
6. Event detection and event marking	-----	18
• Detect movement		
• Detect spindles		
• Detect slow waves		
• Detect eye movements		
• Manually mark events		
7. Saving and exporting your work	-----	25
• Saving your work		
• Exporting your work		
• Closing-up shop		
8. Getting Help	-----	27

Citation

Please remember to acknowledge use of [Counting Sheep PSG](#) by citing the software in your grant applications and publications as:

Ray, L.B., Baena, D., & Fogel, S.M. (2024). "Counting Sheep PSG": EEGLAB-Compatible Open-Source MATLAB Software for Signal Processing, Visualization, Event Marking and Staging of Polysomnographic Data. *Journal of Neuroscience Methods*, 110162.

Terms and conditions

See <https://github.com/stuartfogel/CountingSheepPSG> for details.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above author, licence, copyright notice, this list of conditions, and the following disclaimer.
2. Redistributions in binary form must reproduce the above author, licence, copyright notice, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

COUNTING SHEEP PSG IS INTENDED FOR RESEARCH PURPOSES ONLY.

ANY COMMERCIAL OR MEDICAL USE OF THIS SOFTWARE AND SOURCE CODE IS STRICTLY PROHIBITED.

1. Getting started

• Requirements

- Mac, PC and Linux compatible.
 - MATLAB R2024a or later, including:
 - Signal Processing Toolbox
 - Machine Learning & Statistics Toolbox
 - Designed for use with EEGLAB 2023 or later (<https://eeglab.org>).
1. Download and install EEGLAB using your preferred method:
https://eeglab.org/others/How_to_download_EEGLAB.html
 2. Add the top folder only (*i.e.*, no subfolders) for EEGLAB to the MATLAB path, *e.g.*,
'~/eeglab20xx.x/'
 3. Download and install **Counting Sheep PSG** using one of the following methods:

• Download and installation

Method 1: Download ZIP archive

1. Download the software package from GitHub using:
<https://github.com/stuartfogel/CountingSheepPSG/archive/refs/heads/main.zip>
2. Place the unzipped archive in a location that is accessible to MATLAB.
3. Add the top folder for **Counting Sheep PSG** to the path, *e.g.*,
'~/Documents/MATLAB/countingSheepPSG/'

Method 2: Clone Git Repository

Create a new local copy of a project by retrieving files from Git™ source control using your preferred method. For additional support / details, see: <https://www.mathworks.com/help/simulink/ug/clone-git-repository.html>

1. On the **Home** tab, click **New > Project > From Git**. The **New Project From Source Control** dialog box opens.
2. Enter your HTTPS repository path into the **Repository path** field.
3. In the **Sandbox** field, select the working folder where you want to put the retrieved files for your new project.
4. Click **Retrieve**.

Method 3: Download EEGLAB plugin

Download the EEGLAB plugin and install directly to the '~/eeglab20xx.x/plugins/' directory.

1. Launch EEGLAB by typing 'eeglab' at the command prompt.
2. From the EEGLAB main window, click File > Manage EEGLAB extensions.
3. Locate the 'CountSheepPSG' plugin from the list and click **Install/Update**.
4. After loading an EEGLAB dataset, you can launch **Counting Sheep PSG** via the EEGLAB main window by clicking: **Tools > Counting Sheep PSG**.

- **Software registration**

The first time you launch [Counting Sheep PSG](#), you will be prompted to register the software. [Counting Sheep PSG](#) is free of charge, but you will be asked to provide your first name, last name, your institution/affiliation, your position/title and a valid email address in order to register and use the software.

Alternatively, you can register [Counting Sheep PSG](#) by clicking: **> Help > Register Counting Sheep PSG**.

The license is valid for one computer, for 1 year and is renewable. See the GNU General Public License v3.0 for more information.

Your information is kept secure and private and will never be shared with a third party.

IMPORTANT: By providing these details, the small group of developers of [Counting Sheep PSG](#) can make case for grant support to continue to provide [Counting Sheep PSG](#) for free and to continue to support and develop the software. *Thank you for your support!*

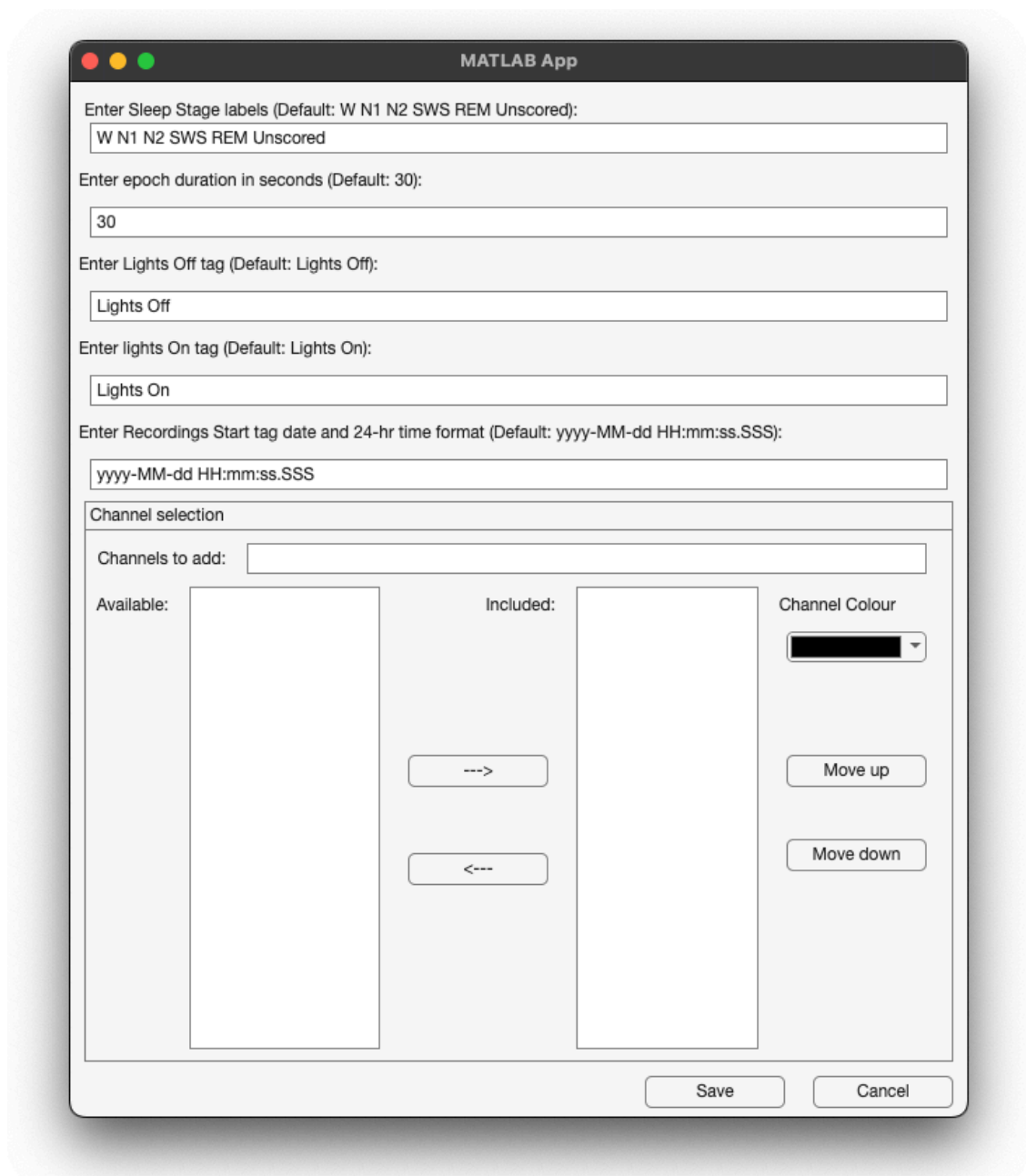
2. Display montage settings

First things first...

Before you can load a new EEGLAB dataset, you'll need to set up a “**display montage**”. This can be done easily by clicking:

[> Counting Sheep > Preferences > Create New / Edit Montage](#)

This will open the “**Create New / Edit Current Study-Specific Display Montage**” window.



From here, you can specify:

- Study-specific sleep stage labels, if any. If none exist in the file already, leave the defaults. If stage scoring labels already exist already in your dataset, they need to be in the EEG.event structure.
- The desired sleep stage scoring epoch duration in seconds (default 30 seconds).
- The “**lights off**” and “**lights on**” event labels, if any. If none exist in the file already, leave the defaults.
- The “**recording start time**” event label date format, if any. If none exist in the file already, leave the default.
- The labels for each channel you wish to display, in the desired order from top-to-bottom.
- The corresponding colours assigned to the traces for the above channel labels.

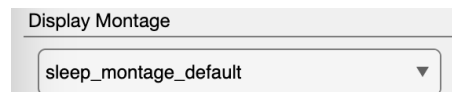
If you are unsure how to find the existing sleep scoring labels, lights on tag labels or how the recording start time marker is formatted, first, load an EEGLAB dataset, and then show the unique event labels; at the MATLAB command prompt, type:

```
>> EEG = pop_loadset; % this will allow you to select an EEGLAB dataset to load  
>> unique({EEG.event.type}) % this will show only unique event labels in your dataset
```

If you are unsure what your channel labels are, at the command prompt type:

```
>> EEG = pop_loadset; % this will allow you to select an EEGLAB dataset to load  
>> {EEG.chanlocs.labels}' % this will show channel labels in your dataset
```

Once completed, you will be prompted to save the montage. This will allow you to load the montage when [Counting Sheep PSG](#) is started, or, to change montages, using the “**Display Montage**” drop-down menu from the toolbar:



If you no longer need a particular montage that has been saved, you can delete it by clicking:

[> Counting Sheep > Preferences > Delete Montage](#)

3. Importing and displaying PSG datasets

- **Importing PSG data to EEGLAB format**

Once you have created a Display Montage, you can load an EEGLAB dataset by using the “open file” toolbar button , or by using the main menu:

> Counting Sheep > Load Dataset

If your PSG data is not already in EEGLAB format, you can use EEGLAB to first import data. See: https://eeglab.org/tutorials/04_Import/Importing_Continuous_and_Epoched_Data.html for more information.

If your data is in either **EDF (*.edf)** or **Brain Products (*.vhdr)** format, **Counting Sheep PSG** supports direct import of these formats and will automatically create a default display montage that you can then edit, as well as create default recording start, lights off and lights on markers.

These functions can be accessed via the main menu:

> Counting Sheep > Import > Import EDF: EEGLAB (Default)
> Counting Sheep > Import > Import EDF: Matlab (Rigorous + Slow)
> Counting Sheep > Import > Import Brain Products

In the case where you have previously sleep stage scored a dataset using **Counting Sheep PSG**, you can load the external *.mat file where this information is automatically backed up after scoring a dataset, accessed by the main menu:

> Counting Sheep > Import > Import Sleep Stages (*.mat)

This function can also be used to import sleep stage scoring done in other software. **Note:** the sleep scoring information *must be in the correct format* for **Counting Sheep PSG** to recognize it. This method is recommended only for advanced users who are familiar with MATLAB data types.

- **Displaying and navigating a dataset**

If you would like to show scale lines (± 37.5 & ± 75.0 μV) in selected channels, click:

> Channels > Show / Hide Scale Lines

If you would like to see the voltage for the data point at the mouse cursor position, you can enable “data cursor mode” . To enable data cursor mode, either use the toolbar button, or click:

> Channels > Turn On/Off Data Cursor Mode

Note: event marking mode cannot be enabled at the same time as “cursor data mode”.

If you would like to hide selected channels, or show selected channels that have already been hidden (note: this does not alter the dataset), click:

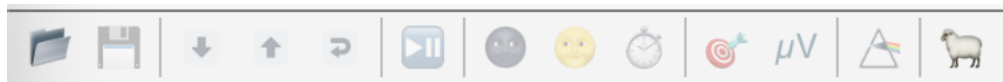
- > Channels > Hide Channels
- > Channels > Show Hidden Channels

Finally, if you wish to increase or decrease the visual scaling of a selected channel (note: this does not alter the dataset), click:

- > Channels > Rescale Selected Channels

You will be prompted to enter a scaling factor. Values between 0 and 1 reduce the scaling by that factor, and values above 1 increase the scaling by that factor. To reset to the default scaling, enter 1.

If you wish to rescale all channels, click the “up arrow” , “down arrow” , or to reset scaling, the “return arrow”  on the toolbar:



You can also use the toolbar to automatically “play” a recording using the “play/pause” button  and to navigate to the epoch where the “light off”  or “lights on”  markers occur. Finally, you can jump to a specific epoch by clicking the “stopwatch” button .

4. Preprocessing

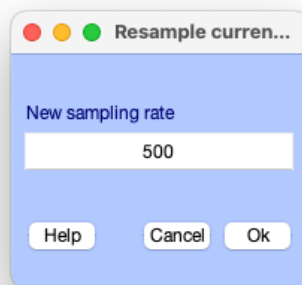
- **Downsample data**

One of the preprocessing steps (depending on your requirements) might be to downsample your data. The major advantage of this is to reduce the file size of the dataset and to increase processing times. This comes at the cost of temporal resolution, and has implications for time-frequency analyses, so great care should be taken when deciding to downsample data. Data can also be “resampled”, typically to match the sampling rate of another dataset to be analyzed in the same manner (*n.b.*, a requirement of certain software packages). Prior to resampling, a low-pass filter at half the resampling frequency is applied to prevent aliasing the data.

The “**Downsample channels**” menu item utilizes the `pop_resample` EEGLAB function and allows the user to resample the dataset (in samples/second) to a different sampling rate. To Resample your data, click:

> Channels > Pre-Process > Downsample Channels

For more information, see: https://eeglab.org/tutorials/05_Preprocess/resampling.html#change-the-sampling-rate



Downsampling can be done on the current dataset, or in batch mode to analyze more than one file at a time. Batch mode can be run without first loading a dataset.

- **Re-reference channels**

Hardware requirement and oftentimes best practices involve recording to a “recording reference” channel, and then, analysis requirements (and often conventions in the literature) necessitate re-referencing to other sites(s).

“**Re-reference channels**” utilizes the `pop_reref` EEGLAB function and allows the user to flexibly convert the dataset to an average reference or re-reference to multiple channels. To re-reference your data, click:

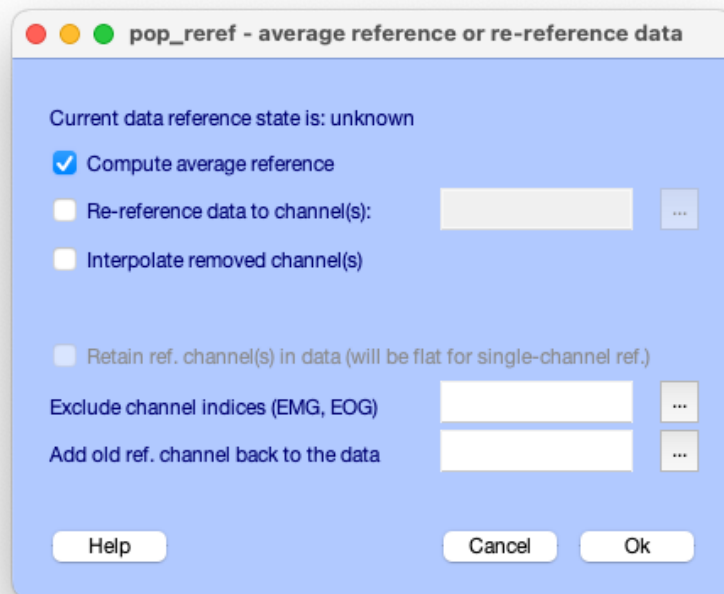
> Channels > Pre-Process > Re-Reference Channels

Optionally, the data can be re-referenced to the average of all channels, by enabling the ‘*Compute average reference*’ checkbox.

Sleep data to be sleep stage scored are normally re-referenced to an average of the mastoid channels. This can be done by enabling the '*Re-reference data to channel(s)*' checkbox and then selecting the 2 mastoid channels (not shown). For PSG data that include EMG, EOG, ECG and other polysomnographic signals, these should be excluded from the EEG re-referencing procedure by selecting them in the '*Exclude Channel indices (EMG, EOG)*' field.

Finally, it is normally recommended to add the original recording reference channel back to the dataset (*n.b.*, this allows subsequent re-referencing of the data, if required) by selecting the '*Add old ref channel back to the data*' checkbox. Note: additional steps are required in order to prepare the dataset for this procedure using the channel editor (see documentation for '*Edit Channels*').

For more information, see: https://eeglab.org/tutorials/05_Preprocess/rereferencing.html



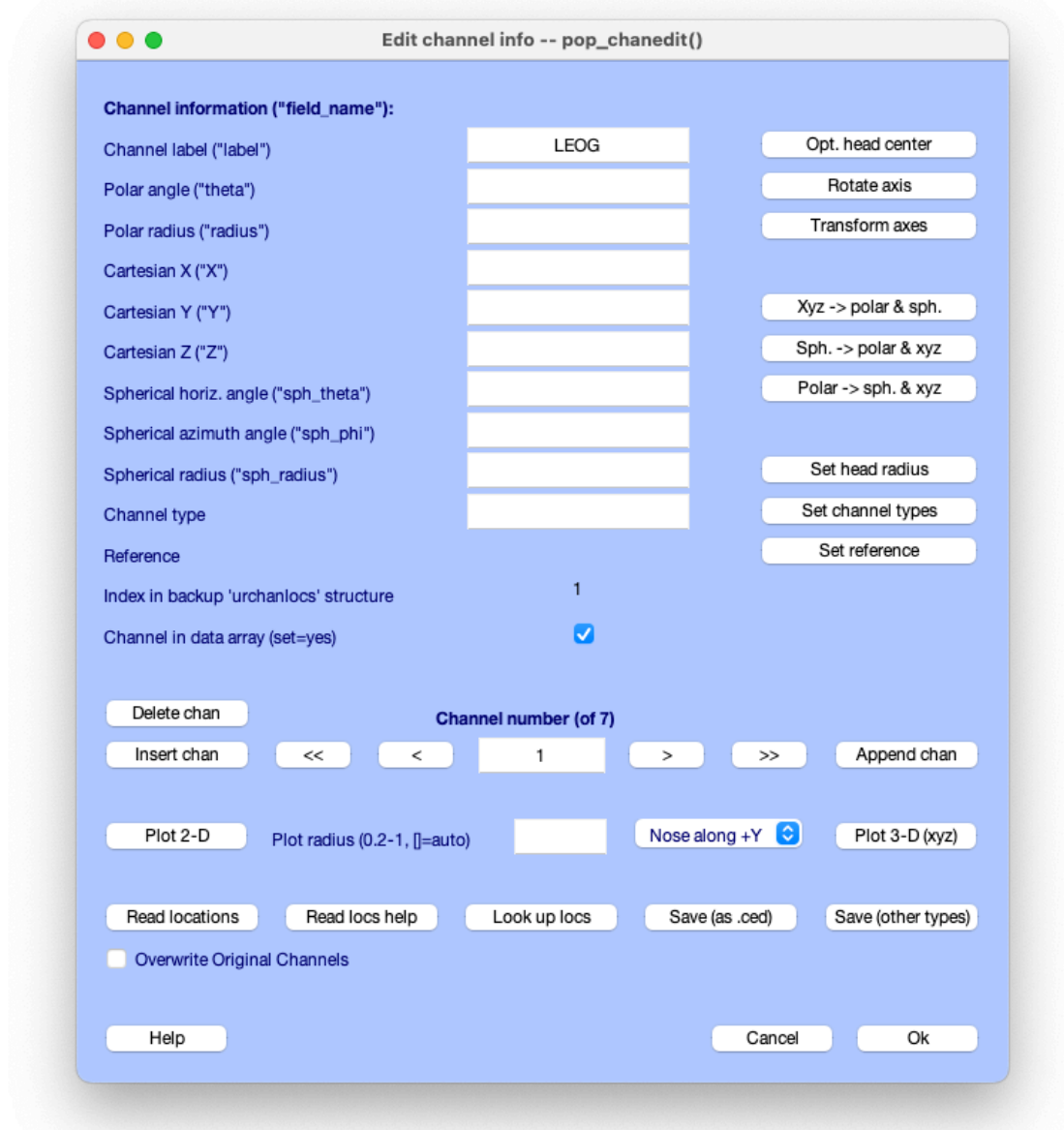
Re-referencing can be done on the current dataset, or in batch mode to analyze more than one file at a time. Batch mode can be run without first loading a dataset.

- **Edit channels**

“**Edit channels**” utilizes the [pop_chanedit](#) EEGLAB function and allows the user to edit channel information and to look up channel locations based on their channel labels (if using standard channel labels) or using EEGLAB format channel location files.

> [Channels](#) > [Edit Channels](#)

In the case that it is not possible to look up channel locations based on their labels alone, or if you have an electrode location file available, press the *Read Locations* button to use the [pop_chanedit](#) EEGLAB function to read in channel locations. If you are unfamiliar with EEGLAB conventions to specify channel locations, press the '*Read locs*' help button for more information.



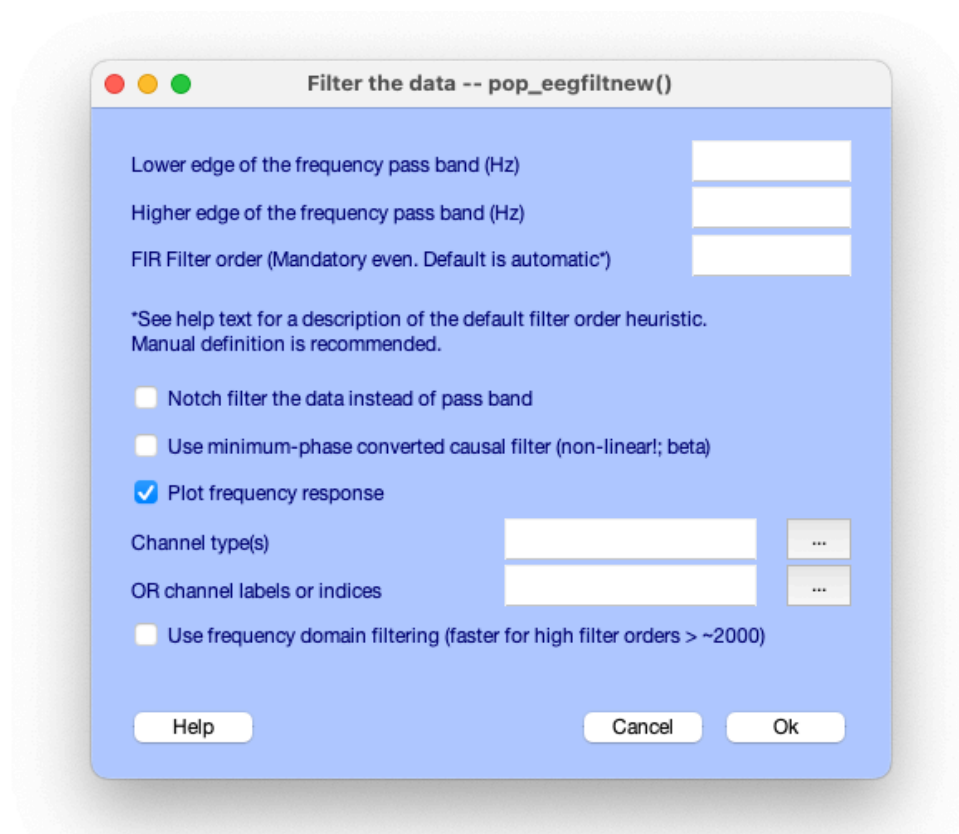
Editing channel information can be done on the current dataset, or in batch mode to analyze more than one file at a time. Batch mode can be run without first loading a dataset.

- **Filter channels**

“**Filter channels**” utilizes the [pop_eegfiltnew](#) EEGLAB function and allows the user to filter data using the latest default EEGLAB filters which use a Hamming windowed FIR filter. To filter your data, click:

> Channels > Pre-Process > Filter Channels

For more information, see: https://eeglab.org/tutorials/05_Preprocess/Filtering.html



Filtering can be done on the current dataset, or in batch mode to analyze more than one file at a time. Batch mode can be run without first loading a dataset.

5. Artifact correction and signal processing

- **Mark bad channels**

“**Mark bad channels**” creates an event to identify one or more channels as “bad data” and can be used to exclude data from an entire channel from subsequent analyses.

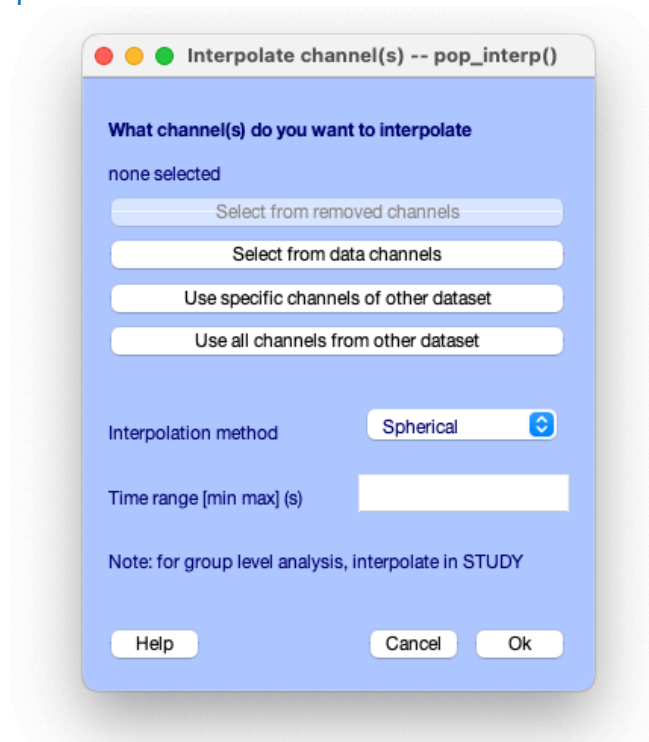
> Channels > Mark Channel as “Bad Data”

- **Interpolate bad channels**

“**Interpolation of missing or bad channels**” utilizes the `pop_interp` EEGLAB function which allows the user to replace one or more missing or bad channels with interpolated data. After specifying the ‘bad’ channel(s) to interpolate and the channel(s) to be included in the interpolation, you will be prompted to optionally enter the exact time to start the interpolation. Leave this field blank to interpolate the selected channels for the entire duration of the recording. Interpolation is done using the spherical (default) method for interpolation.

To interpolate a bad channel, click:

> Channels > Interpolate Bad Channel



For more information, see: https://eeglab.org/tutorials/05_Preprocess/Filtering.html

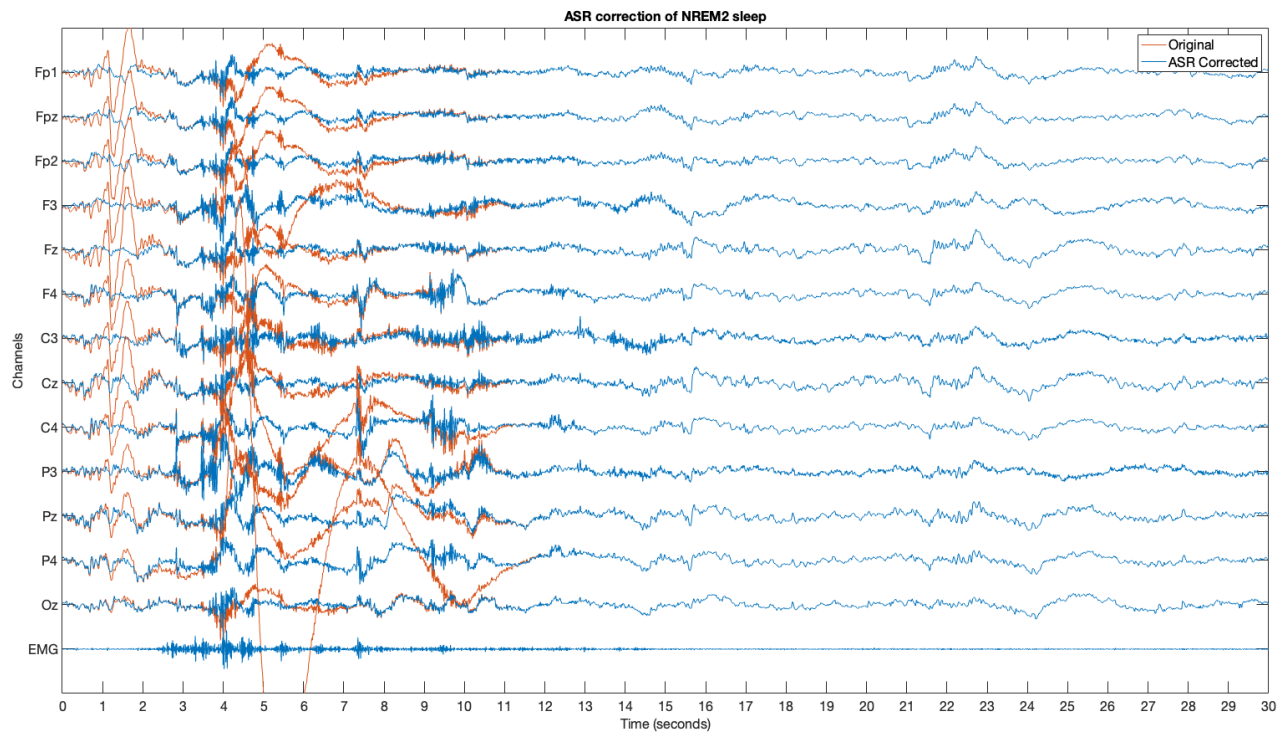
- **Artifact subspace reconstruction (ASR)**

“**Artifact subspace reconstruction**” utilizes the `clean_asr` EEGLAB function from `clean_rawdata` plugin. This function allows the user to specify the sleep stages of interest to be applied as well as the standard deviation cutoff value (default 30). To run Artifact Subspace Reconstruction, click:

> Channels > Artifact Subspace Reconstruction

After selecting the relevant channels and sleep stages, ASR is applied segment-by-segment to each identified sleep stage period. This approach ensures tailored artifact removal while preserving critical EEG characteristics unique to different sleep stages. The user can review and retain the ASR-processed data.

For more information, see: https://eeglab.org/tutorials/06_RejectArtifacts/cleanrawdata.html



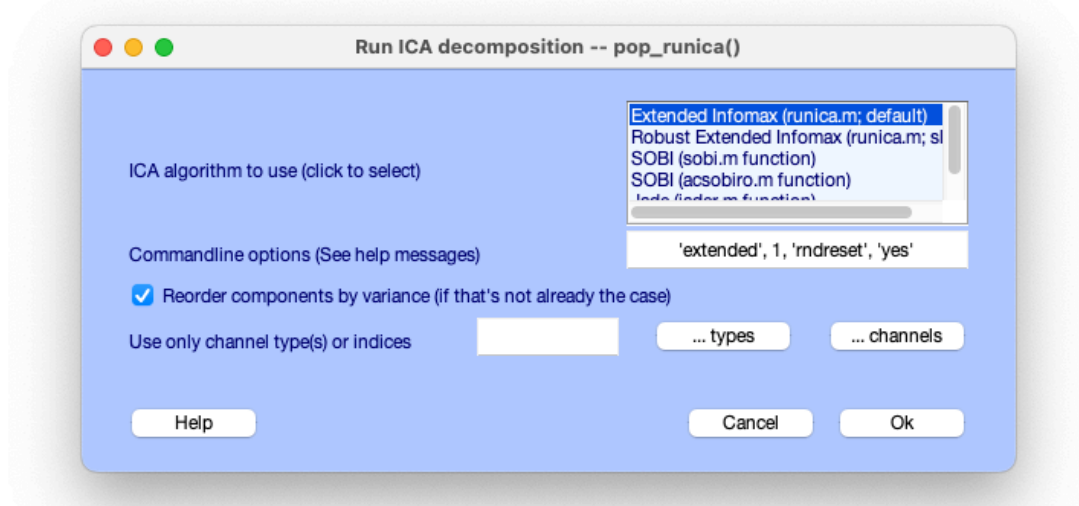
- **Independent component analysis (ICA)**

“Independent component analysis” utilizes the `pop_runica` EEGLAB function and allows the user to display the data in component space, permitting the removal/subtraction of artifacts embedded in the data (*e.g.*, muscle artifact, eye blinks, etc.).

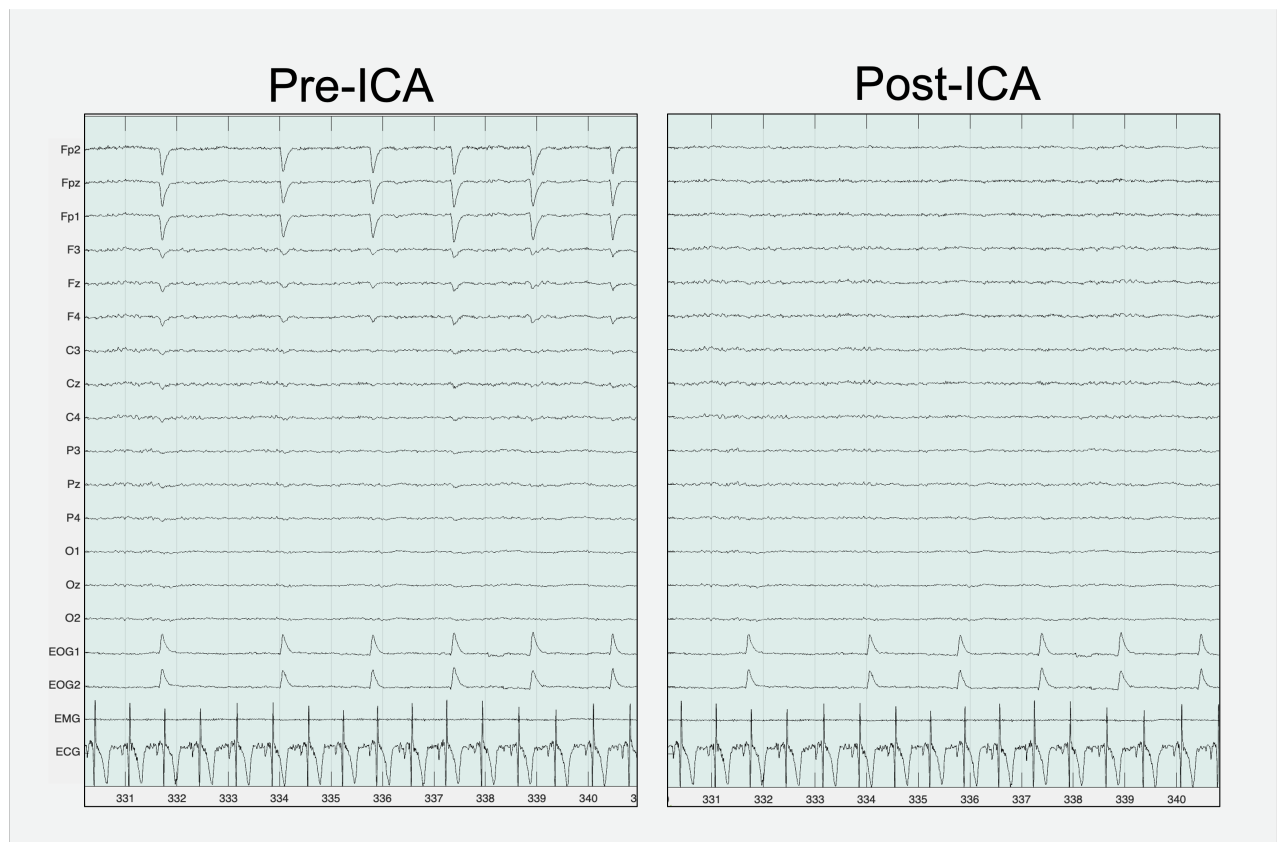
For more information, see: https://eeglab.org/tutorials/06_RejectArtifacts/RunICA.html.

To run independent component analysis, click:

- > Channels > Independent Component Analysis
 - > Run ICA
 - > Show / Hide ICA Components
 - > Remove Selected Components



The independent component analysis (ICA) implemented in [Counting Sheep PSG](#) features several ICA decomposition algorithms to choose from, however Infomax ICA using [runICA.m](#) is the default algorithm and is recommended by EEGLAB. The user also has the option to reorder the components by variance (if that's not already the case) and to select the channel type(s) or indices that will be used in the analyses. Once the analysis is complete, the user has the option to show/hide the ICA components and to remove the selected components.



- **Power spectral analysis**

Power spectral analysis” utilizes the [pop_spectopo](#) EEGLAB function and allows the user to calculate the power spectrum and calculates the power spectrum density of the EEG using the Welch method.

The Welch approach is advantageous as it provides a better estimation of the signal as it is less sensitive to noise in the signal and is not impacted by non-stationarity of the EEG. The approach implemented in **Counting Sheep PSG** has been specifically adapted to sleep data analysis. The key adaptations include parameters that align with and support the analysis of sleep data, typically scored in 30-second epochs. Specifically, we use a default 5-second Hamming window, resulting in six windows per 30-second sleep stage epoch. This achieves a frequency resolution of 0.2Hz, ideal for capturing defining features of sleep, including frequencies as low as 0.4Hz. Averaging periodograms over these windows significantly reduces variance, catering to the non-stationarities inherent in sleep EEG. Output is in dB.

Plotting options include: mean spectral plot per sleep stage, time-frequency spectrogram with hypnogram, and topographic plots. The resulting output and display of the results categorized according to sleep stage labels. Plots can be generated with or without opening a dataset after running the power spectral density analysis.

For more information, see:

https://eeglab.org/tutorials/08_Plot_data/Plotting_Channel_Spectra_and_Maps.html

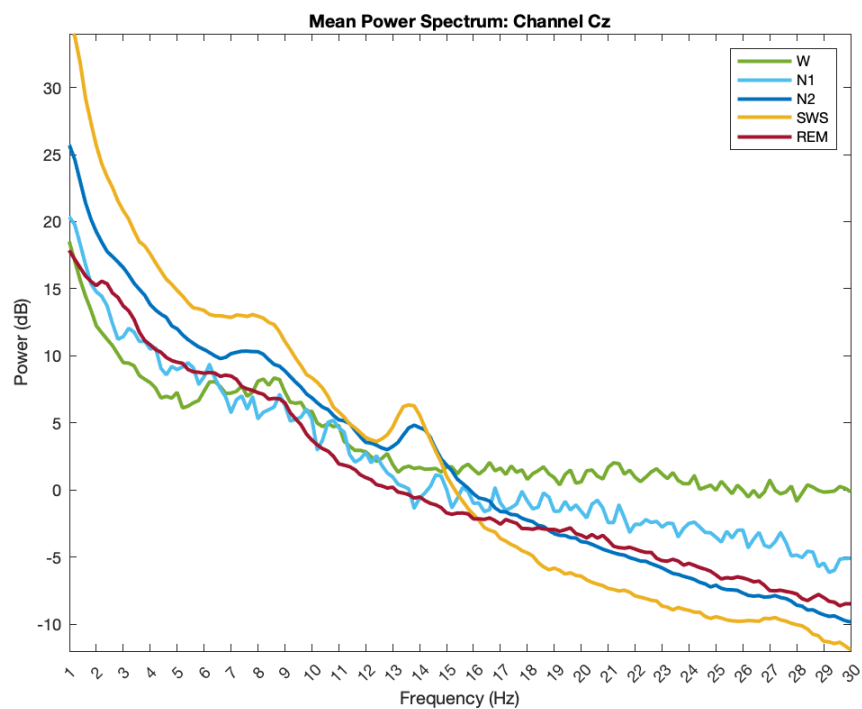
To run power spectral analysis and plot the output, click:

- > Channels > Power Spectral Analysis
 - > Run Power Spectral Density (FFT – Welch)
 - > Plot Power Spectrum Density
 - > Plot Time-Frequency Spectrogram
 - > Plot Topography

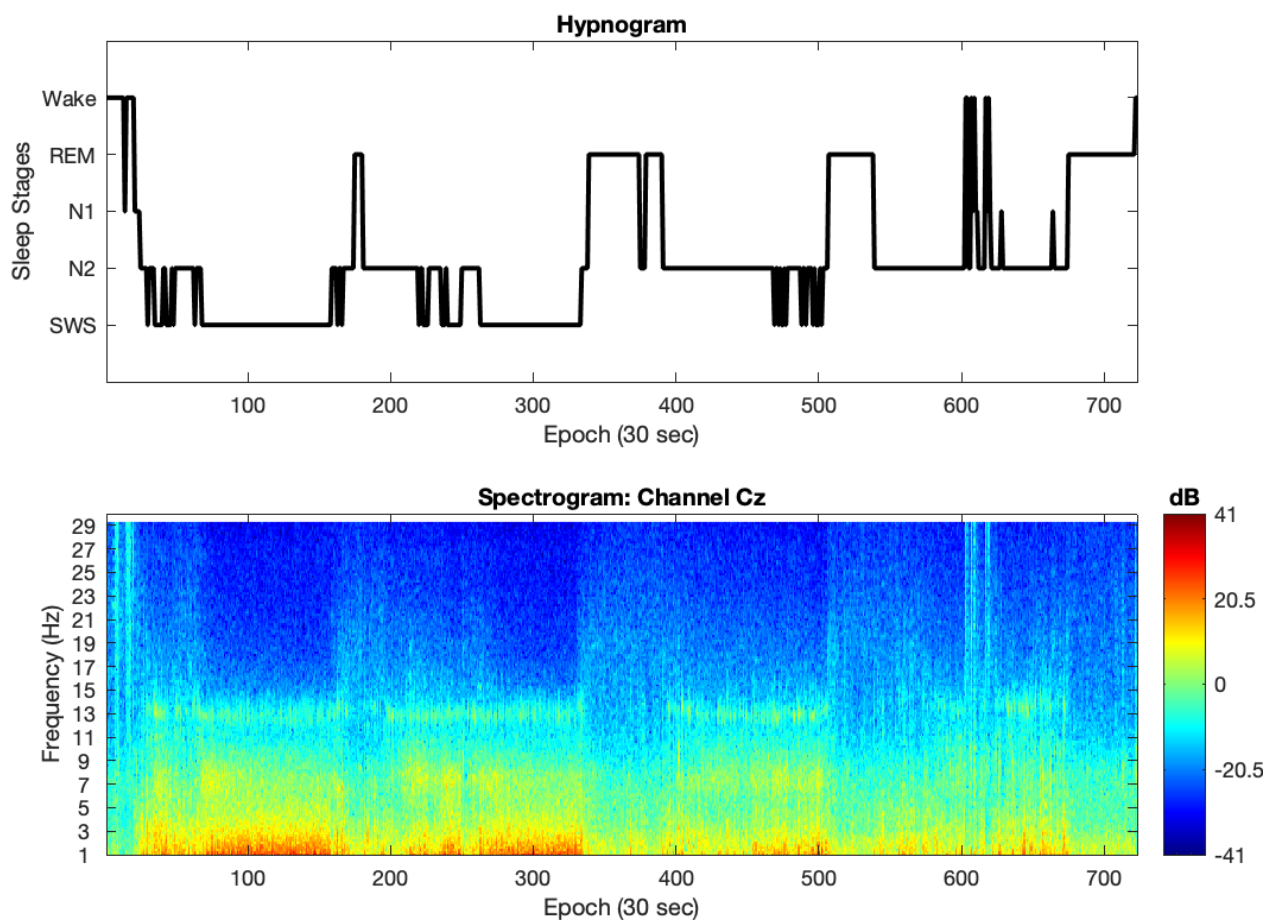
The definable parameters of the power spectral analysis are:

- channels to include
- minimum and maximum frequencies of the analyses
- event labels of the 'bad data' to be excluded from the analyses
- channels to include in the analyses

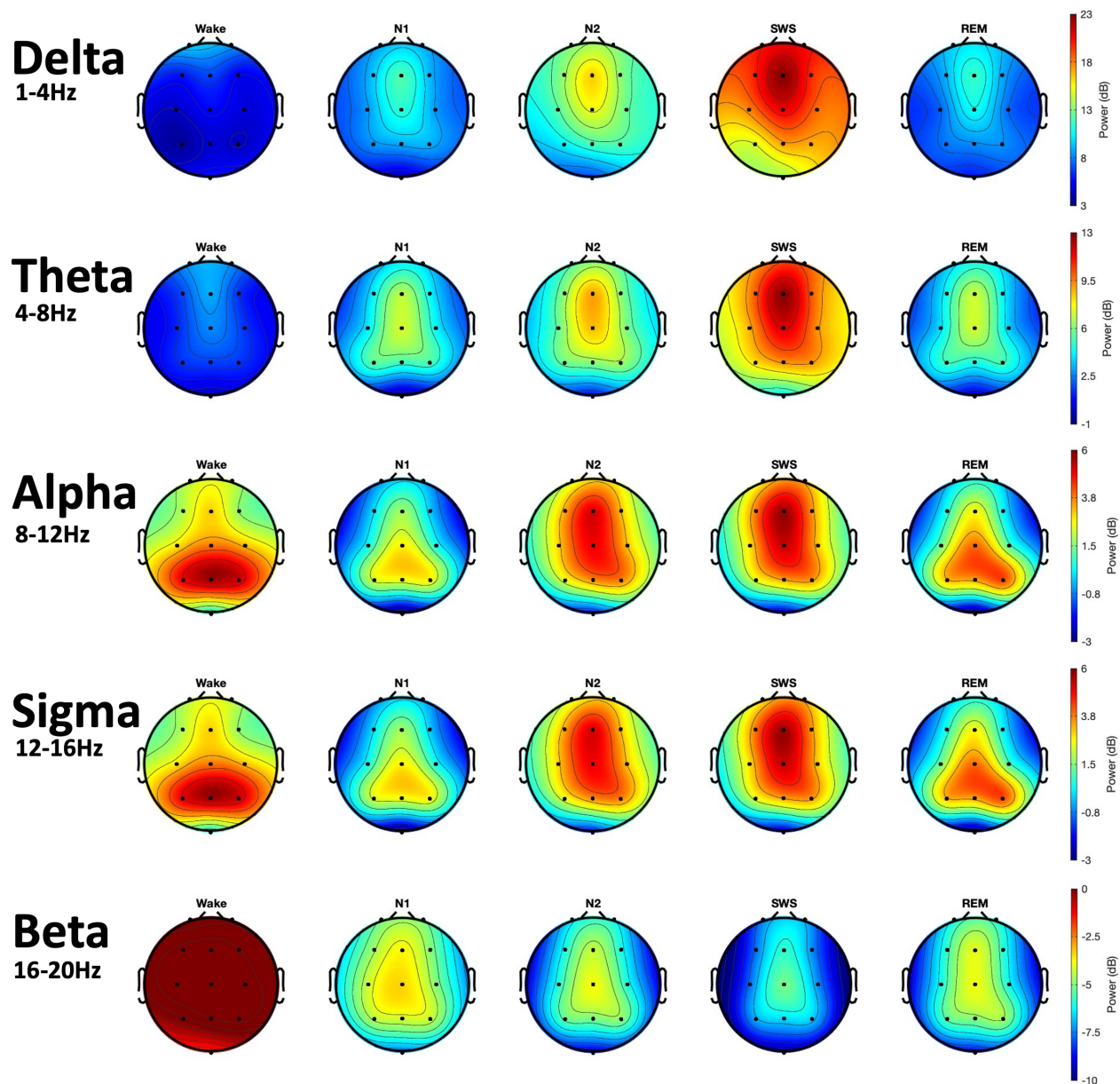
Once the analysis is complete, the user has the option to plot the power spectrum density, time-frequency spectrogram and topography. The power spectrum density plot displays the mean power spectrum of a specific channel, per sleep stage, for the entire recording.



The spectrogram displays the time-frequency representation of a specific channel for the entire recording along with the hypnogram to allow for easy visual identification of global changes in the spectral properties of the signal which are often associated with changes in sleep stages.



Finally, the topographic maps depict the mean power values computed from the entire recording, at a specified frequency, displayed per sleep stage.



6. Event detection and event marking

Movement artifacts, sleep spindles, slow waves and rapid eye movements can be automatically detected, employing well-established and validated approaches.



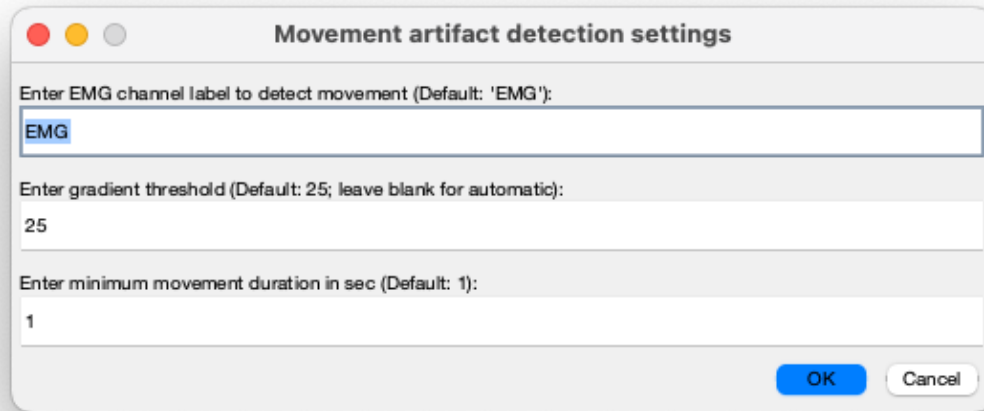
- **Detect movement**

Movement artifact detection employs a simple algorithm that transforms the selected EMG signal into the first derivative to identify rapid changes in the amplitude of the EMG signal (*i.e.*, movements) according to a predefined delta threshold (default: 25 μ V). The change in the EMG signal must be sustained over a predefined period (default: 1 sec). If so, the traces are marked with a movement artifact event across all channels.

Note: it is important to visually verify the accuracy of the detection, as EMG signal quality can easily impact the detection.

To run automatic movement artifact detection, click:

[> Events > Detect Movement](#)



Movement artifact detection settings

Enter EMG channel label to detect movement (Default: 'EMG'):

EMG

Enter gradient threshold (Default: 25; leave blank for automatic):

25

Enter minimum movement duration in sec (Default: 1):

1

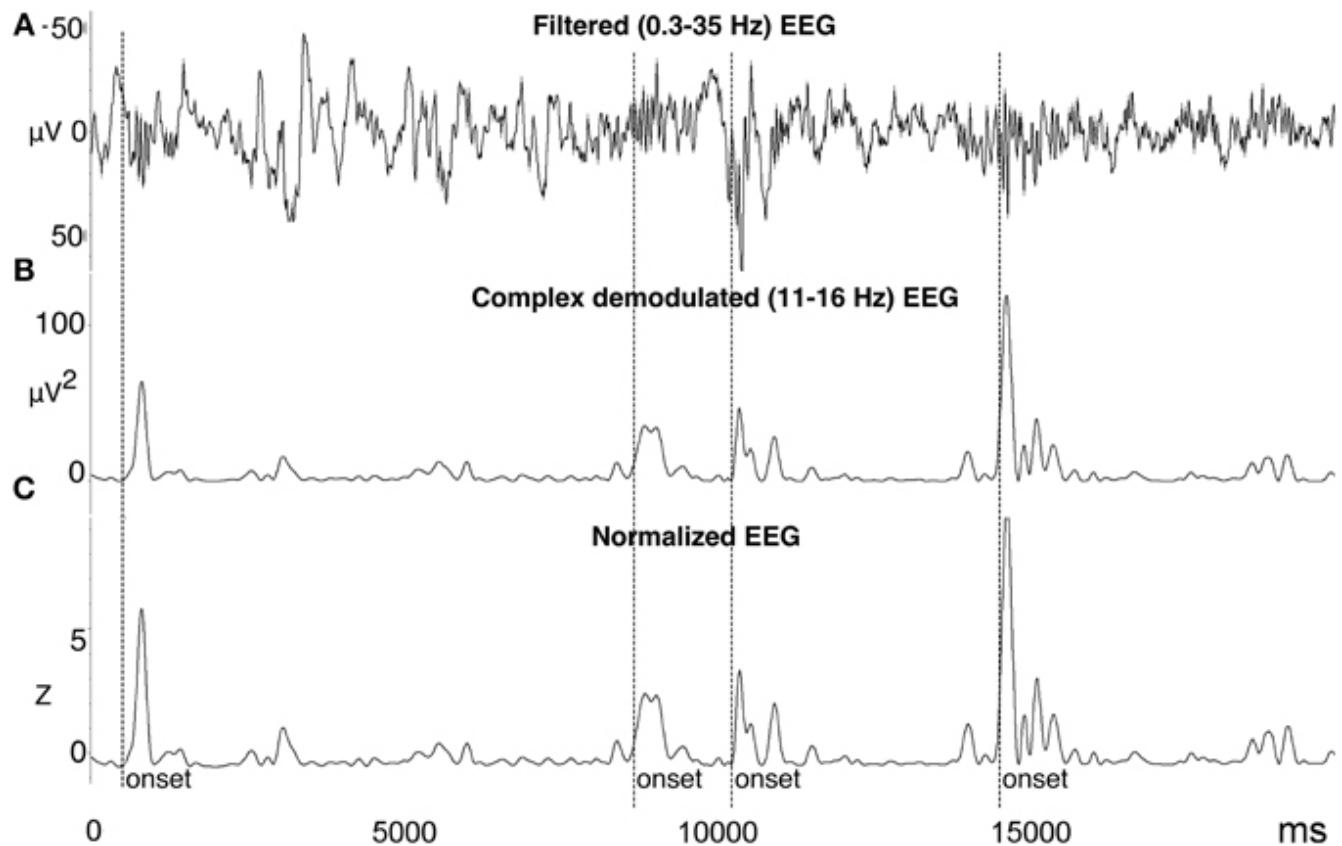
OK Cancel

- **Detect spindles**

Automatic sleep spindle detection employs a validated and well-established approach that involves several steps: First, the EEG is transformed using complex demodulation (CD), producing a new time series of instantaneous magnitude (μV^2) in the frequency band of interest (e.g., 11–16 Hz). Then, the CD time series is normalized to Z-scores, calculated from a window the size of the current contiguous sleep stage (e.g., uninterrupted Stage 2 or Stage 3 sleep). Spindle onsets are then detected when Z exceeds a predefined threshold (default: $Z > 2.3$, i.e., 99th percentile). To accurately measure the entire length of the spindle, the onset is adjusted to be the first point at which $Z = 0.5$ prior to the amplitude threshold Z , and the offset as the first point at which $Z = 0.5$ after the amplitude threshold Z .

This approach has been validated and compared to multiple expert visual scoring and non-expert crowd-sourced scoring. For details see:

Ray, L., Sockeel, S., Soon, M., Bore, A., Myhr, A., Stojanoski, B., Cusack, R., Owen, A., Doyon, J. & Fogel, S. (2015). Expert and crowd-sourced validation of an individualized sleep spindle detection method employing complex demodulation and individualized normalization. *Frontiers in Human Neuroscience*, 9(507), 1-16. <https://doi.org/10.3389/fnhum.2015.00507>



Sleep spindles can be detected in any sleep stage, on any channel, and excluded during user-defined artifact events, with customized event labels (*e.g.*, 'Movement'). The frequency range for sleep spindles can be specified, as well as the amplitude threshold, minimum duration, maximum duration and minimum inter-spindle interval. Either complex demodulation (default) can be employed, or, root mean square, if preferred.

To run automatic sleep spindle detection, click:

[> Events > Detect Spindles](#)

Spindles can be detected on the current dataset, or in batch mode to analyze more than one file at a time. Batch mode can be run without first loading a dataset.

Detect Spindles -- detect_spindles()

Event Labels

Label for spindle event: Spindle

All sleep stage labels (comma-separated): W, N1, N2, SWS, REM, Unscored

Sleep stages to detect spindles (comma-separated): N2, SWS

Movement artifact label (comma-separated): Movement, Arousal

File suffix for new dataset: SpDet

Complex demodulation (CD) / Root mean square (RMS) settings

☒ Use CD (default, checked) or RMS (unchecked)

Central frequency for CD: 13.5

Bandwidth about CD central frequency: 5

High pass filter for RMS (optional): 11

Low pass filter for RMS (optional): 16

Z-Score normalization and detection settings

Z-Score event detection threshold: 2.3

Minimum duration (sec) between spindle events: 0.25

Minimum spindle event duration (sec): 0.5

Maximum spindle event duration (sec): 3.0

Channel Options

Channel labels or indices: Fz Cz Pz ...

Help Cancel Ok

A number of spindle parameters are automatically exported to Microsoft Excel (*.xlsx) and MATLAB data files (*.mat). Automatically exported parameters include: sleep spindle latency (data points), duration (data points), channel, peak (data points), peak amplitude (μV), integrated amplitude (*i.e.*, 'area': $\mu\text{V/srate}$), frequency (Hz) and sleep stage.

- **Detect slow waves**

The automatic slow wave detection method employs period amplitude analysis (PAA) and is adapted from well-established, previously published methods (Bersagliere and Achermann, 2010). The PAA

approach measures slow waves directly without any complex signal transformations, thereby it inherently has near-perfect face validity.

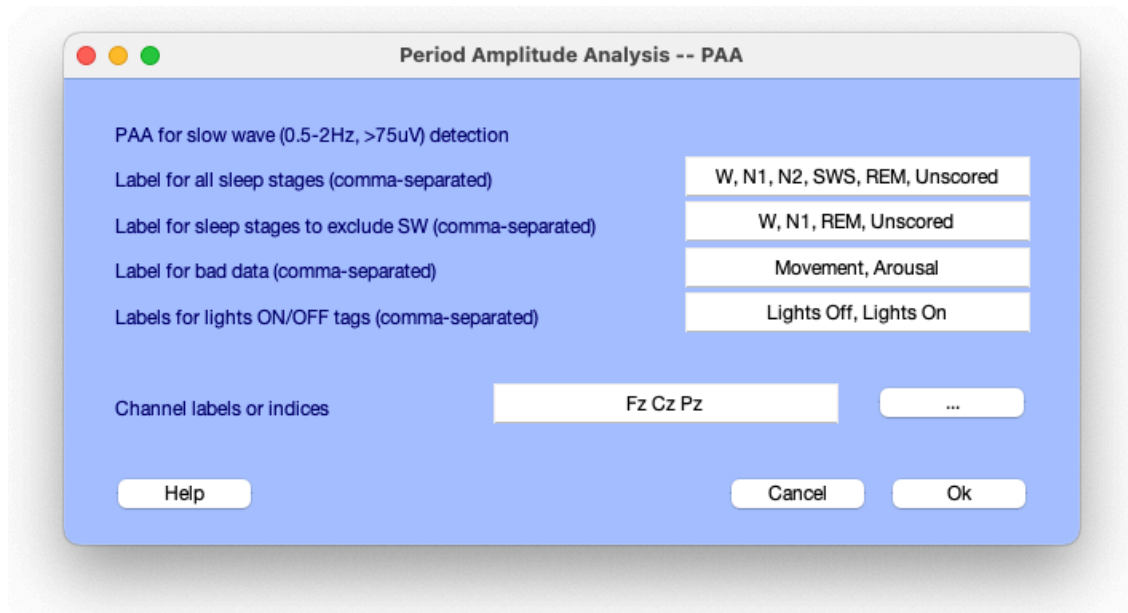
First, the EEG signal is band-pass filtered (32nd order Chebyshev type 2 low-pass filter, 80-dB stopband attenuation, 2.15 Hz frequency cut-off; 64th-order Chebyshev type 2 high-pass filter, 80-dB stopband attenuation, 0.46 Hz frequency cut-off). The cut-off frequencies were selected to achieve minimal attenuation in the band of interest while keeping a good attenuation of the neighboring frequencies. The filters are applied in the forward and reverse directions to achieve zero-phase distortion.

Next, half waves are determined as negative or positive deflections between two consecutive zero crossings in the band-pass filtered signal. In line with scoring rules for slow waves, the peak-to-peak amplitude threshold of $75\mu\text{V}$ (Rechtschaffen and Kales, 1968) is applied to each half wave and its neighbouring half wave.

To run automatic slow wave detection, click:

[> Events > Detect Slow Waves](#)

Slow waves can be detected on the current dataset, or in batch mode to analyze more than one file at a time. Batch mode can be run without first loading a dataset.



A number of slow wave parameters are automatically exported for slow wave half-waves to Microsoft Excel (*.xlsx) and MATLAB data files (*.mat).

Automatically exported parameters for each half-wave include: slow wave latency (data points), duration (seconds), channel, peak (data points), peak amplitude (μV), integrated amplitude (*i.e.*, 'area': $\mu\text{V/srate}$), average amplitude (area/duration), frequency (Hz) positive slope, negative slope and sleep stage.

- [Detect rapid eye movements](#)

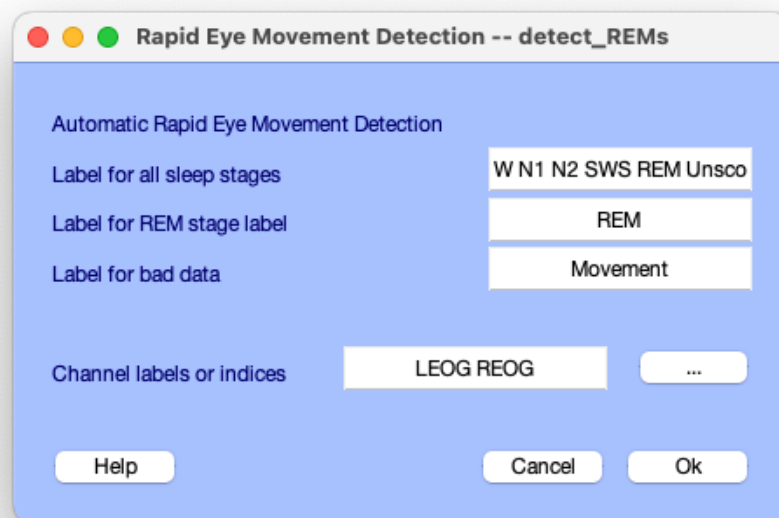
The rapid eye movement (EM) detection algorithm is based on an established approach using a matched filtering algorithm adapted from Yetton et al's, 2016 validation study. This approach is most commonly known in the field as the method (Hatzilabrou et al., 1994) that was incorporated into Stellate Harmonie software (Stellate Systems, Inc., Montreal, Canada).

EOG signals from REM sleep are filtered from 0.5 to 10 Hz (default) and smoothed using a Hamming window. The filtered EOG signals are divided into consecutive 1-s windows. EMs are then identified by their correlation using the magnitude of the squared correlation with a template available in Yetton et al., 2016, of a typical EM for the left and right EOG signals. Only monocular EMs above the amplitude threshold of $23\mu\text{V}$ are included in the detection. Finally, only conjugate EM events, *e.g.*, left and right monocular EMs occurring in the same time window, are then marked on the original EOG trace at the peak (maximum) amplitude of the EM.

To run automatic rapid eye movement detection, click:

[> Events > Detect Eye Movements](#)

Rapid eye movements can be detected on the current dataset, or in batch mode to analyze more than one file at a time. Batch mode can be run without first loading a dataset.



- **Manually mark events**

One of the most useful set of features allows the manual marking of events directly on the traces. Events can be marked on one or all traces with custom event labels, deleted or merged across one or more epochs. To enable “**event marking mode**”, click the ‘bullseye’ icon  on the toolbar, or click:

[> Events > Turn On / Off Event Marking Mode](#)

One event marking mode is enabled, the event marking tools will be available, where you can specify a custom event label, whether to mark events across all channels, increase the window size to multiple epochs (useful for marking long events that span multiple epochs), or advance the epoch

window by 1 second to mark an event that straddles the epoch boundary. The delete and merge events buttons are always enabled.

Note: event marking mode cannot be enabled at the same time as “cursor data mode”.

Events can be single-point events, by clicking the trace at the point of event insertion, or, events can have a duration specified by click-and-dragging. In both cases, events can be added to one or all channels by toggling the “**All Channels**” checkbox.



All events of a certain type can be deleted from the trace by clicking:

[> Events > Delete Events](#)

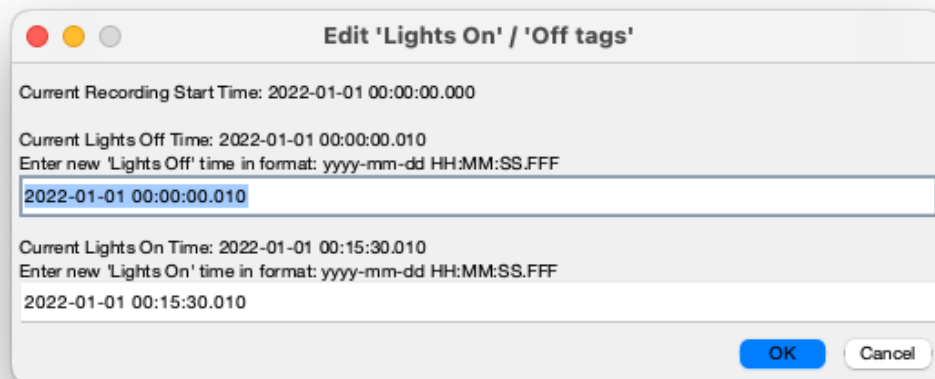
To show or hide events of a certain type, click:

[> Events > Hide Events](#)

[> Events > Show Hidden Events](#)

Finally, the latency of the ‘Lights On’ and Lights Off’ tags can be edited by clicking:

[> Events > Edit ‘Lights On’ / ‘Lights Off’ Tags](#)



7. Saving and exporting your work

- **Saving your work**

Any changes to PSG datasets can be saved to the existing file by clicking:

[> Counting Sheep > Save Dataset](#)

Alternatively, changes to an existing dataset can be saved to a new dataset with a different name or location by clicking:

[> Counting Sheep > Save Dataset As...](#)

- **Exporting your work**

One or more *.mat files that contain a backup of the raw sleep scoring information can be selected to generate a sleep report that is exported to Microsoft Excel (*.xlsx) format for easy analysis of sleep architecture data.

To generate a sleep report from one or more datasets, click:

[> Counting Sheep > Generate Sleep Report](#)

ID	recStartTime	recStopTime	lightsOFFtime	lightsONtime	TRT	TSWP	TST	SOL	SE	NA	WASO	N1latency	N2latency	SWSlatency	REMLatency	Unscored	WAKE	N1	N2	SWS	REM	N1percent	N2percent	SWSpercent	REMLpercent
SS_01	2024-05-01 23:25	2024-05-02 7:27	2024-05-01 23:33	2024-05-02 7:09	482	455.5	445	2.5	97.6	16	12.5	2.5	6	20	131.5	0	11.5	15	258	91	81	3.37	57.93	20.47	18.22
SS_02	2024-08-12 22:46	2024-08-13 6:16	2024-08-12 22:49	2024-08-13 6:17	450	447	426	13	95.1	15	12	12.5	18.5	28	102	0	22	11	258	94.5	63	2.47	60.52	22.21	14.81
SS_03	2022-04-12 23:55	2022-04-13 7:21	2022-04-12 23:59	2022-04-13 7:21	446	442	418	13	94.5	12	16	12.5	18.5	27.5	95	0	24.5	19	200	106	94.5	4.43	47.73	25.24	22.61
SS_04	2023-04-30 23:18	2023-05-01 7:04	2023-04-30 23:26	2023-05-01 7:04	466	457.5	443	2	96.8	12	15.5	2	5.5	13.5	85.5	0	15	9.5	253	82.5	98.5	2.14	57	18.62	22.23

In addition, the information contained in the EEG.event structure can be exported to provide all the raw event information marked on the dataset that is available in EEGLAB. A summary of these events is also calculated in the report exported to Microsoft Excel (*.xlsx).

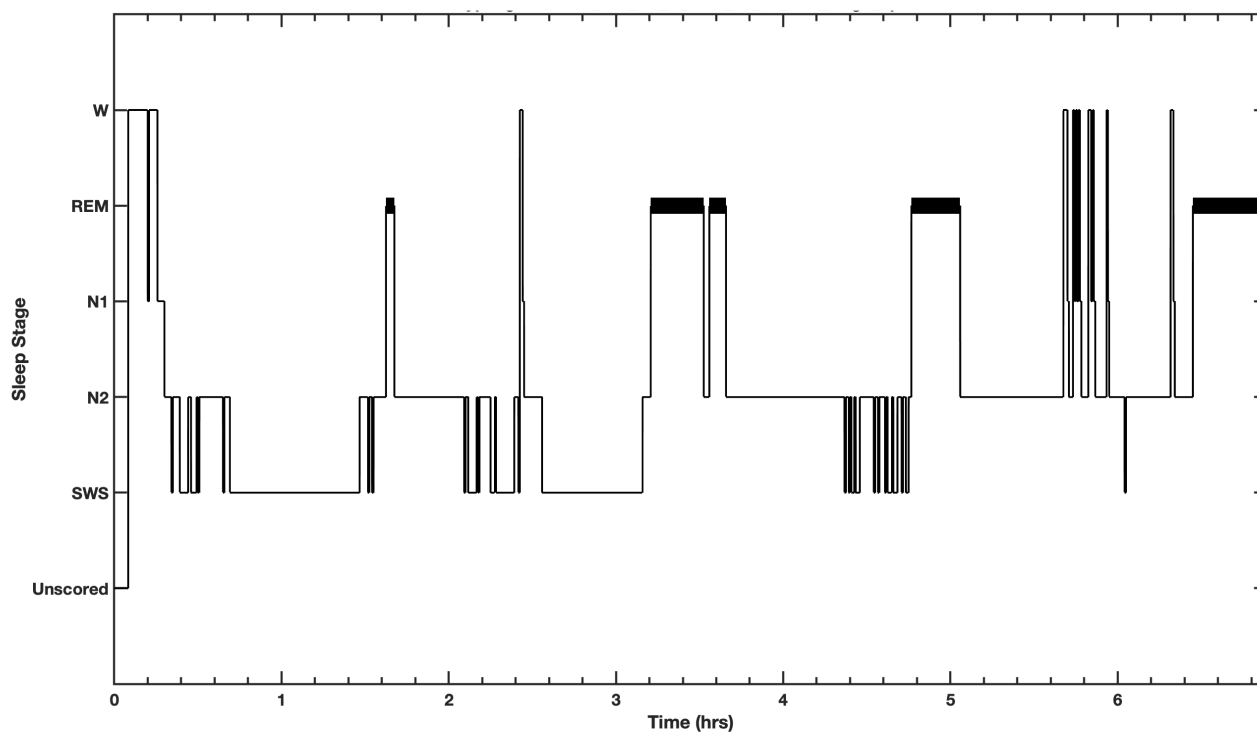
To generate an event report, click:

[> Counting Sheep > Generate Event Report](#)

Publication-quality MATLAB figures of sleep stage scoring hypnograms can be generated, and then saved to any file format supported by MATLAB. By default, a *.png image file is saved to the default directory.

To generate hypnogram, click:

[> Counting Sheep > Generate Hypnogram](#)



- **Closing-up shop**

Finally, datasets can be closed without exiting, by clicking:

[> Counting Sheep > Clear Dataset](#)

To exit the program, click:

[> Counting Sheep > Quit Counting Sheep](#)

In both cases, a prompt will appear if there are unsaved changes.

8. Getting help

To open this file help through the menu system, click:

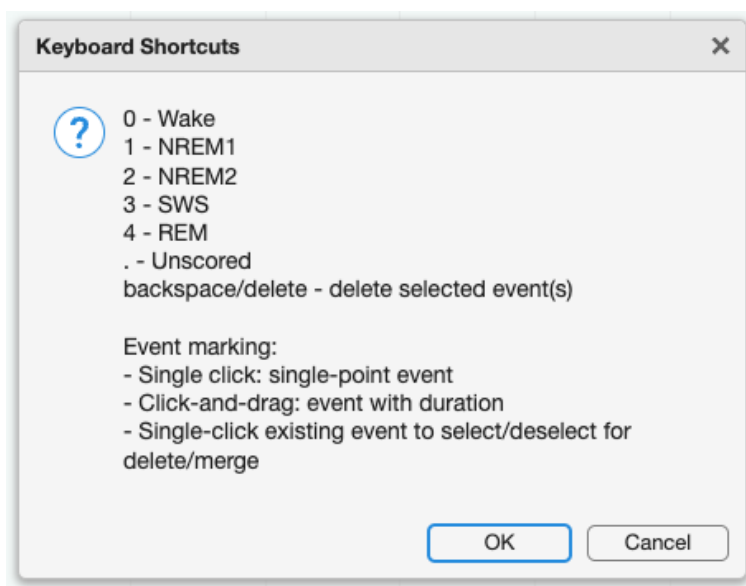
[> Help > Open User Manual](#)

On certain operating systems (*e.g.*, MacOS, Linux) and configurations, updates from Git can be automatically downloaded and installed. To update via git, click:

[> Help > Check for Updates on Git \(Mac Only\)](#)

There are a number of useful keyboard shortcuts for sleep stage scoring and navigation. A list of these shortcuts can be found by clicking:

[> Help > Keyboard Shortcuts](#)



For information about [Counting Sheep PSG](#), it's authors, and how to cite the software, click:

[> Help > About Counting Sheep PSG](#)