UNIVERSITÉ LIBRE DE BRUXELLES,
UNIVERSITÉ D'EUROPE

ULB

SpikeDet

SPM EEG Spike detection toolbox

Manual

TABLE OF CONTENTS

1 INTRODUCTION

SpikeDet, a SPM EEG Spike detection toolbox, allows to detect spikes in an EEG record by using a fully automated method described in [Nonclercq2012] and based on [Nonclercq2009]:

We propose a fully automated method of interictal spike detection that adapts to interpatient and intrapatient variation in spike morphology. The algorithm works in five steps. (1) Spikes are detected using parameters suitable for highly sensitive detection. (2) Detected spikes are separated into clusters. (3) The number of clusters is automatically adjusted. (4) Centroids are used as templates for more specific spike detections, therefore adapting to the types of spike morphology. (5) Detected spikes are summed.

Detected spikes are marked as "spike" event with a value corresponding to the electrode name where the spike has been detected. At the end of the detection process, it also computes and prints the spike index defined by the ratio between the number of one second window containing at least one spike and the total length of the EEG record in seconds.

This manual gives you a guideline to use the EEG Spike detection toolbox (version 1) which was implemented in SPM version 12 (SPM12) [Ashburner2013]. The spike detection algorithm is detailed in [Nonclercq2012] and is based on [Nonclercq2009]. Let us notice that three Matlab toolboxes are needed to use SpikeDet:

- Matlab Statistic Toolbox to use K-mean function.
- Matlab Signal Processing Toolbox to find the signal peaks.
- Matlab Image Processing Toolbox to compute the normalized correlation between two signals.

In future versions of this toolbox, the functions used by the last two toolboxes will be probably re-implemented. Matlab R2013a was used for the development of this toolbox.

Please also refer to the SPM user manual [Ashburner2013].

If you use this toolbox for a publication (in a journal, in a conference, etc.), please cite it by including as much information as possible from the following:

> *Antoine Nonclercq & Rudy Ercek, SpikeDet : a SPM EEG Spike detection toolbox, Université Libre de Bruxelles, http://beams.ulb.ac.be/research-projects/spm-eeg-spike-detection-toolbox, 2016.*

Please also cite both related publications: [Nonclercq2012] and [Nonclercq2009].

2 LICENSE

As SPM, the license attached to this toolbox is GPL v2, see https://www.gnu.org/licenses/gpl-2.0.txt. From https://www.gnu.org/licenses/gpl-2.0.html, it implies:

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

3 SPM AND SPIKEDET TOOLBOX INSTALLATION

1. Download SPM12 Toolbox on the website : http://www.fil.ion.ucl.ac.uk/spm/
2. Decompress the SPM12 zip file (spm12.zip) in a folder of your choice.
3. Run Matlab and add the SPM folder to Matlab search path: Menu "HOME" and button "Set Path" in the last Matlab versions. Another solution: go to the SPM folder and type the command "addpath(pwd)" each time you want to run SPM in Matlab.
4. Download the SpikeDet toolbox on the website: http://beams.ulb.ac.be/research-projects/spm-eeg-spike-detection-toolbox
5. Decompress the zip file in the SPM folder. The "spm_eeg_artefact_spikes.m" file and the "spikes" folder should both be present in the SPM folder.
6. If you are using Matlab with Linux or MAC OS, you have to compile the 2 "c files" in the directory "spikes" by typing the command "mex *.c" in this directory.

4 CONVERSION FROM EDF (OR OTHER EEG) FILE TO SPM FILE

1. Run SPM by typing the "spm" command in Matlab; a window with 6 buttons appears.
2. Click on the "M/EEG" button; three windows (figures) appear, among which the "Menu" window shown in Figure 1.
3. In the list box "Convert" (1) of the Menu window (Figure 1), select and click "Convert" (scroll this window to select it!); the "Select M/EEG data file" window shown in Figure 2 appears.
4. In this window, select the EEG file you want to convert and click on "Done" (1); another window with "M/EEG data conversion" shown in Figure 3 appears.
5. This window (Figure 3) asks to the user to "Define Settings". You should answer "Yes" (1) (the "Just Read" button (2) could also work depending on your case, see below).
6. The "Batch Editor" window for "Conversion" module now appears (see Figure 4).
7. It presents several options. The most interesting option of this window is the "Check trial boundaries" (5) which should be changed from "Yes" to "No" in order to be able to correctly use the spike detection algorithm. Indeed, in some cases, the EDF file contains a boundary (break) each second, so the converted signal is split in 1 sec trials (the number of trials is equal to the number of seconds in the signal). In this case, the spike detection algorithm cannot work properly. If the EEG file is too large (e.g. EEG channels cannot be easily displayed, display is slow, Matlab crashes when trying to display EEG channels …), other options can help to reduce its size: change "Reading mode" "Continuous" (1) from "Read all" (2) to "Time Window" in order to take only a sample from the signal (e.g. from 0 to 1000 sec), change channel selection (3) from "All" to e.g. "Custom Channel". These options can also be used with another module called "Crop" in the "Preprocess" window Menu (see (2) in Figure 1) before or after spikes detection. By default, a prefix "spmeeg_" is added to the original filename after conversion,

you can change the output filename by inserting a name in (5). Figure 5 presents the "Batch Editor" with some of these options modified.

8. After configuring the options of the "Batch Editor", you can start the conversion by clicking on the "play" button (6) of Figure 4. A SPM EEG file is created in the current Matlab directory (the current Matlab directory has to be writable!); a SPM file is composed of two files: a matlab file (.mat) containing the header information and a data file (.dat) containing the EEG samples.

Please refer to the SPM manual for further information about converting EEG files [Ashburner2013].

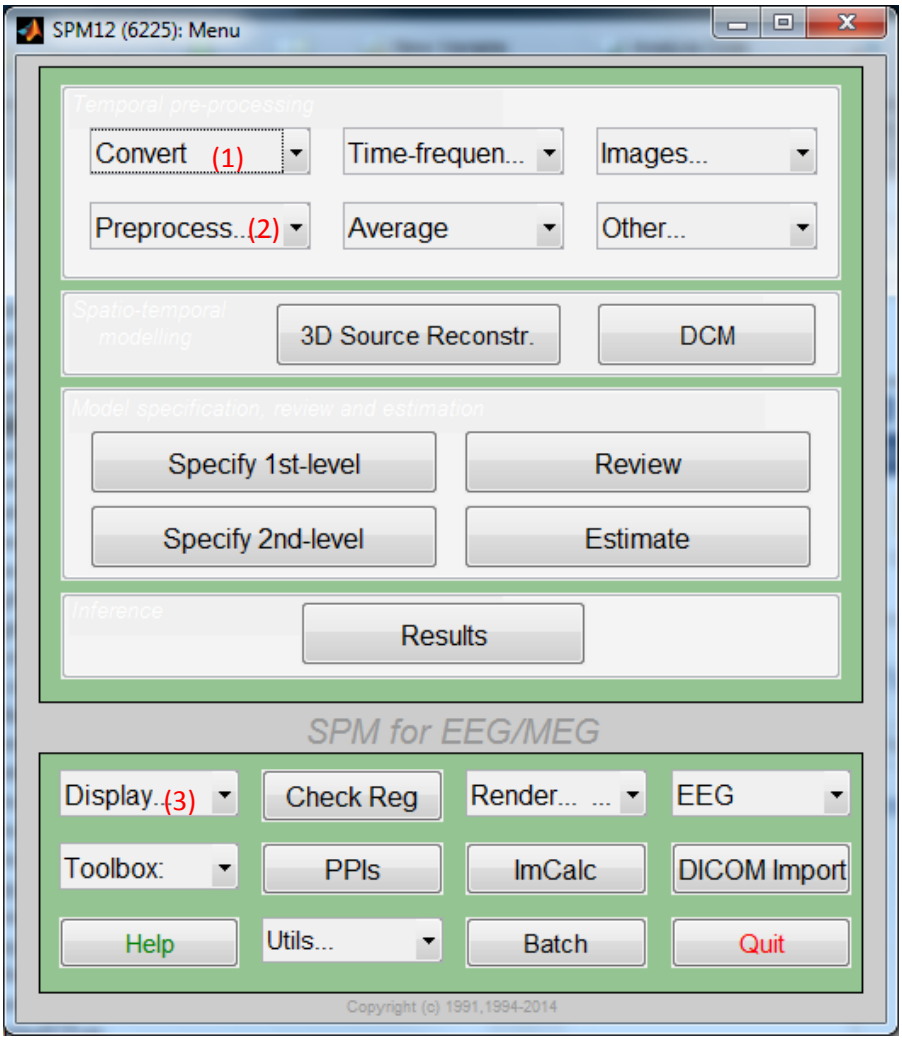


Figure 1 : SPM Menu

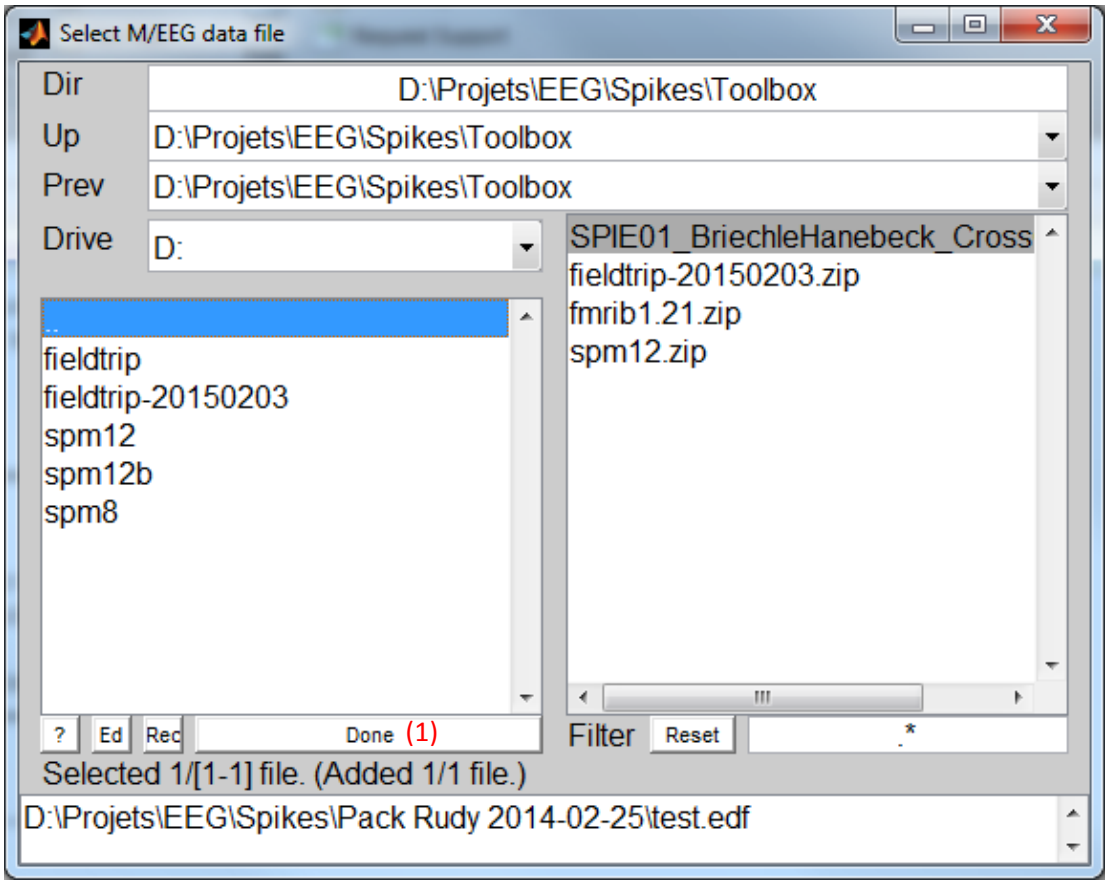


Figure 2: Select M/EEG data file window for file conversion

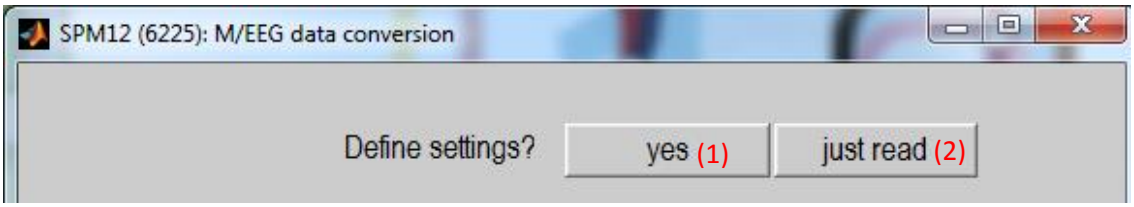


Figure 3: M/EEG data conversion window in order to "Define Settings"

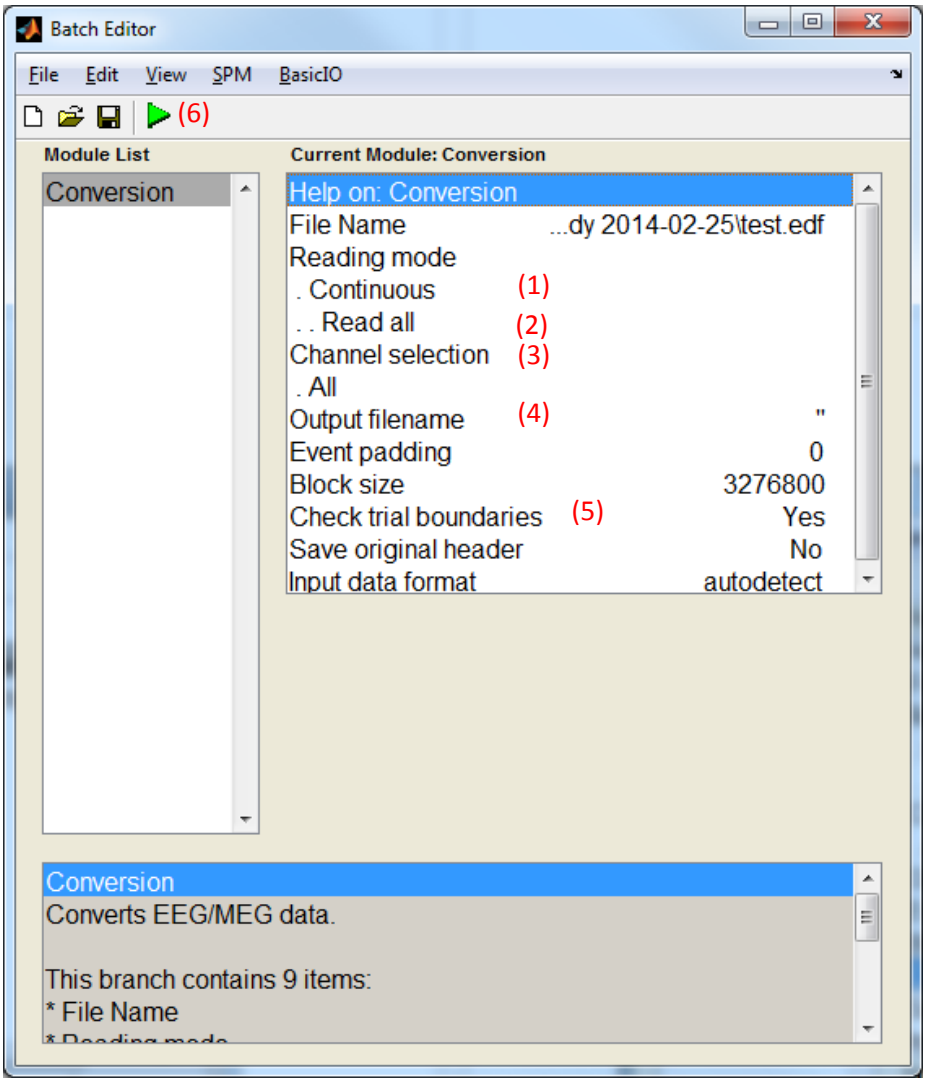


Figure 4: Batch Editor for EEG file conversion with default options

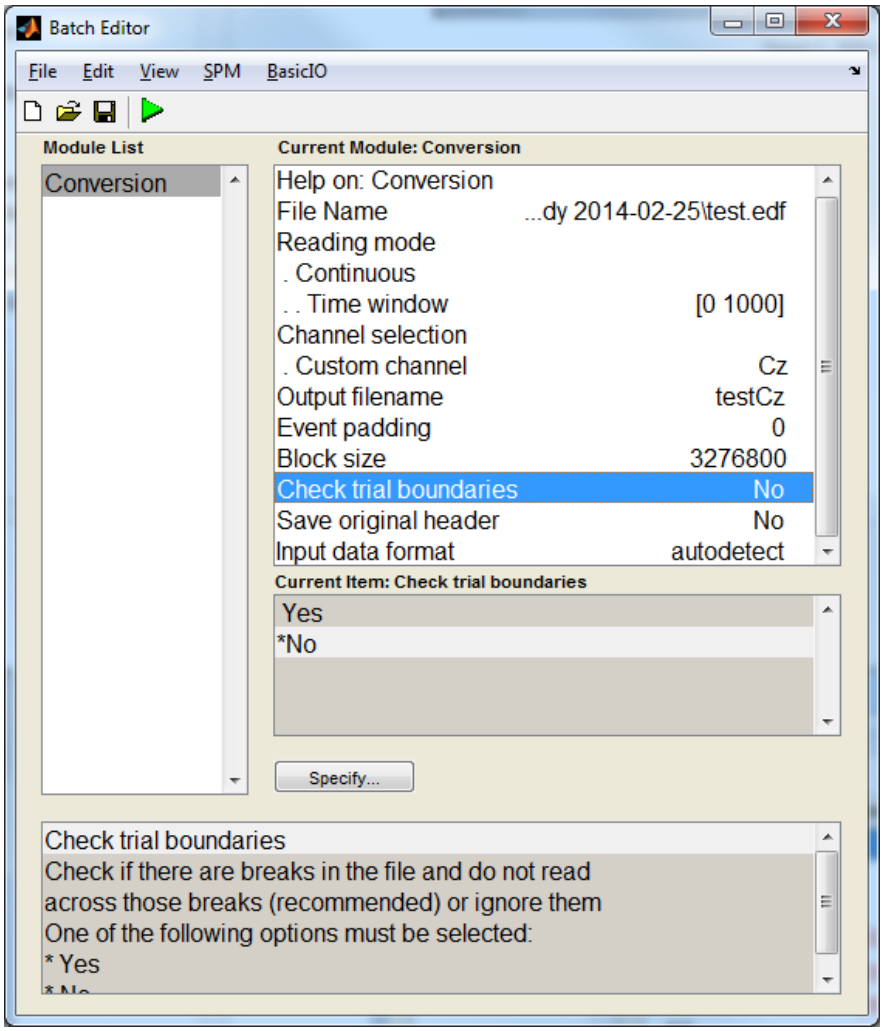


Figure 5: Batch Editor for EEG file conversion with options changed

5 DISPLAY THE SPM (EEG) FILE

A SPM EEG file can be displayed in the "Graphics" window shown in Figure 6. To do so, you have to select the "M/EEG" option in the "Display …" list box (3) of the SPM menu (see Figure 1) and select the SPM header file (.mat) you want to display by using the window shown in Figure 2. When an EEG file is directly displayed after conversion, all channels are often plotted in the tab "OTHER" (1) of the "Graphics" window. Indeed, most channel types are not automatically recognized after conversion. If necessary, they can be specified later (see next section).

Another interesting tab is the "info" tab (2) where you can have some information about the EEG file. Let us notice that in most cases, the trials correspond to events and if breaks/boundaries were detected in the original EEG file, they are considered as trials. You can see 2 trials of this type in the EEG plots of Figure 6 (vertical green lines). Several buttons can be used in the "Graphics" window to change the plot configuration (e.g. for time (6) and voltage amplitude (7)). Please refer to the SPM user manual for more information about these controls [Ashburner2013].

Let us notice that, in most cases, the tabs "MEG", "MPLANAR" and "MCOMB" of the "Graphics" window are empty. The tab "EEG" gives the plot of all EEG channels if the type is specified.

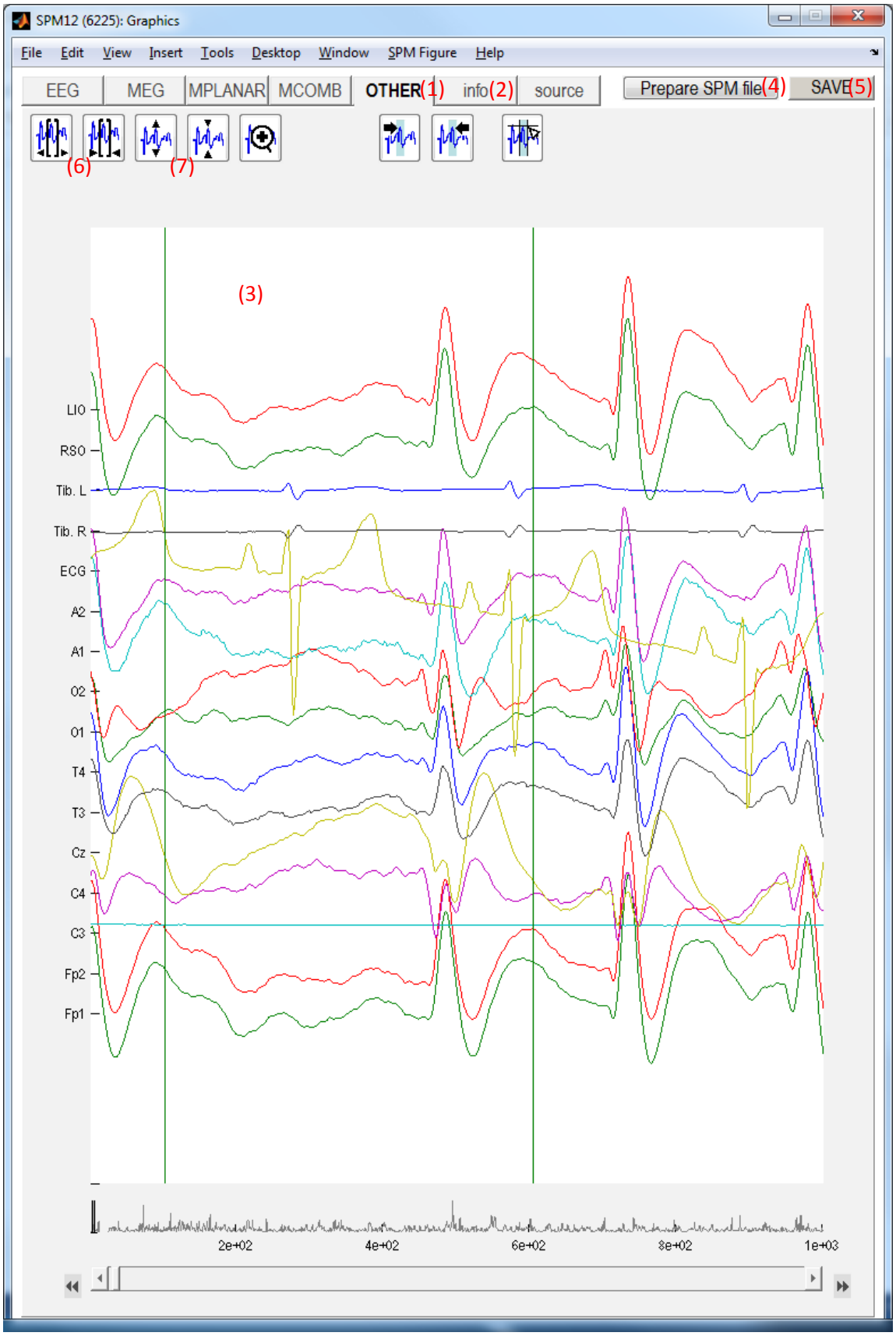


Figure 6: Display of a SPM EEG file in the "Graphics" window

6 PREPARE THE SPM (EEG) FILE

After converting an EEG file, it is really interesting to prepare the SPM EEG file even if it is not necessary before using the EEG spike detection toolbox. Indeed, you can specify the different types of electrodes and even give a position for them. In this section, we will present one method to prepare a SPM EEG file. Other methods could be also used (see [Ashburner2013]).

The different steps to prepare our converted SPM EEG files are the following ones:

1. To open the "M/EEG prepare" window, two methods can be used :
    a. From the "Graphics" window shown in Figure 6, click on the button "Prepare SPM file" (4); the "M/EEG prepare" window shown in Figure 7 appears. You should click the "OK" button (1) when the file preparation is finished.
    b. In the "Convert" list box (1) of the SPM "Menu" window (see Figure 1), select the "Prepare" item; the "M/EEG prepare" window shown in Figure 8 appears. In the menu of this window, a new "File" item (1) is present. You can open an SPM EEG File by clicking on the "Open" item. The window shown in Figure 2 then allows to select the file.
2. You can specify the type of channels by using the "Channel types" (3) menu of the "M/EEG prepare" window and by clicking the type of channel you want to specify. For example, if you select "EEG", the window "Set type to EEG" shown in Figure 9 proposes to select the channels you want to configure as "EEG". Once the EEG channels have been selected, click the OK (1) button. Let us notice that you have to keep the "Ctrl" key pushed to select more than one channel (i.e. a list box item).
3. Once the EEG channels are set, you can specify the position of each EEG electrode by using the menu item "Sensors" (4) of the "M/EEG prepare" window. In this menu, the most straightforward method is to select the "Assign default" item in the "Load EEG Sensor" submenu which assigns a default position for each electrode based on the electrode name.
4. In the "2D projection" menu (5) of the "M/EEG prepare" window, you can check and change the electrode position by clicking on "Edit existing EEG" or "Project 3D (EEG)". The window shown in Figure 10 allows to change them by clicking on an electrode (the selected electrode is green) and just then by clicking on the new position. When the position is changed, click on "Apply" in the "2D Position" menu (5).
5. You can also change the montage by using the "Define EEG referencing" submenu in the "Sensors" menu (4) (see SPM user manual [Ashburner2013] for more information).
6. Once the preparation is finished, you should save all your modifications. Depending on the method you used to open the SPM file, you have to select the corresponding method below:
    a. Click on the "OK" button (1) of the Figure 7. The "Save" button (5) color of Figure 6 becomes red (see Figure 11) and you have to click on it in order to save all the modifications you have done on this file.
    b. In the "File" menu (1) of Figure 8, you have to click on "Save" in order to save all modifications (preparation) you have done on the SPM file.

Figure 11 shows the Graphics window after preparation is complete. The "EEG" tab now includes all the EEG channels. If you switch to the "info" tab, you have all information about this file (see Figure 12) and in the sub. tab. "Channels", you can check the type of each channel. Moreover, the "show sensors 3D positions" button allows to display all channel positions in 3D; an example of 3D electrode positions is shown in Figure 13.

After preparation, an interesting EEG visualization is the scalp interpolation based on the 3D projection of electrode positions. To plot it, click on the button (1) of Figure 11. Figure 14 shows an example of a scalp interpolation for an epileptic spike. The temporal location is given by a vertical black line (2) shown in Figure 11. In this figure, let us remember that the vertical green lines are the 1 sec. trials (events).

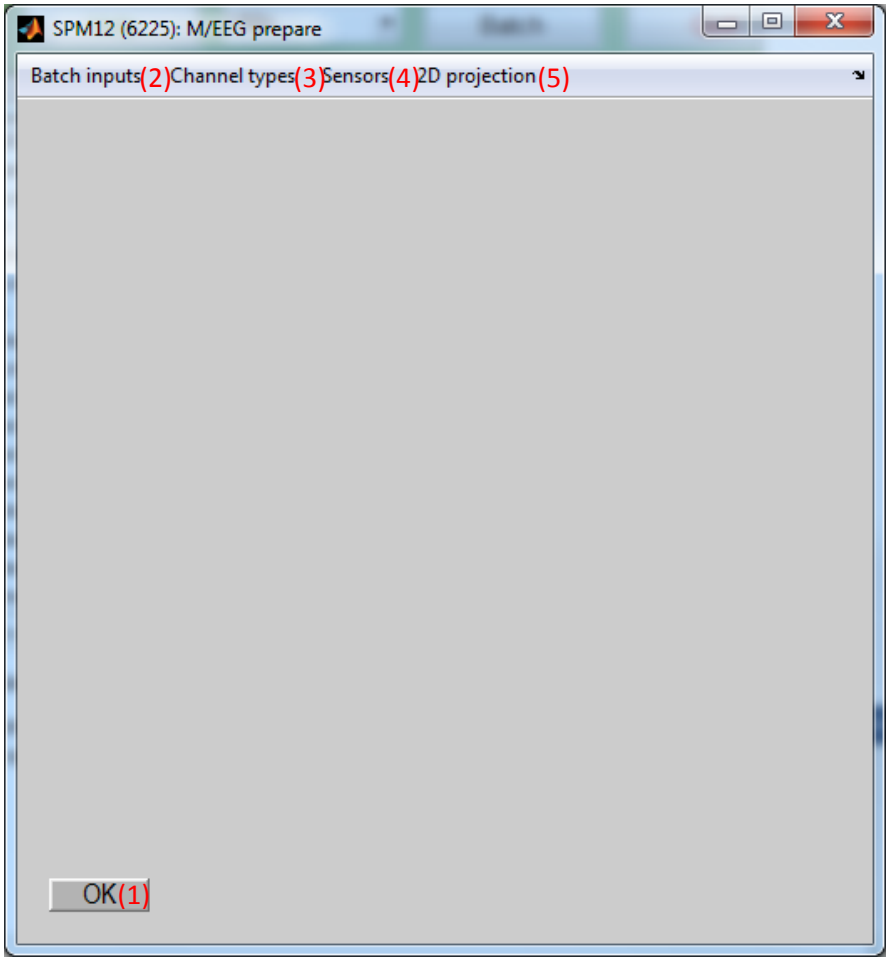


Figure 7: "M/EEG Prepare" window after clicking on "Prepare SPM file" button of the "Graphics » window

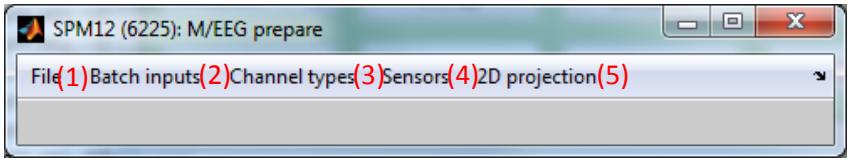


Figure 8: "M/EEG Prepare" window after clicking on "Prepare" of the list box "Convert" of the SPM Menu

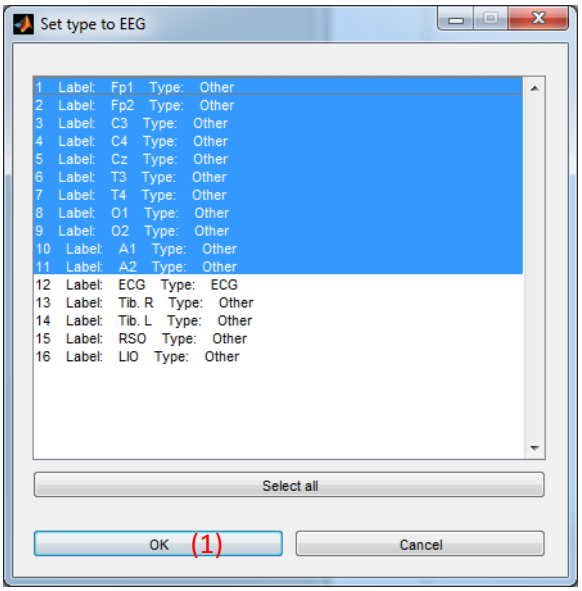


Figure 9 : "Set type to EEG" for some channels when preparing a SPM EEG file

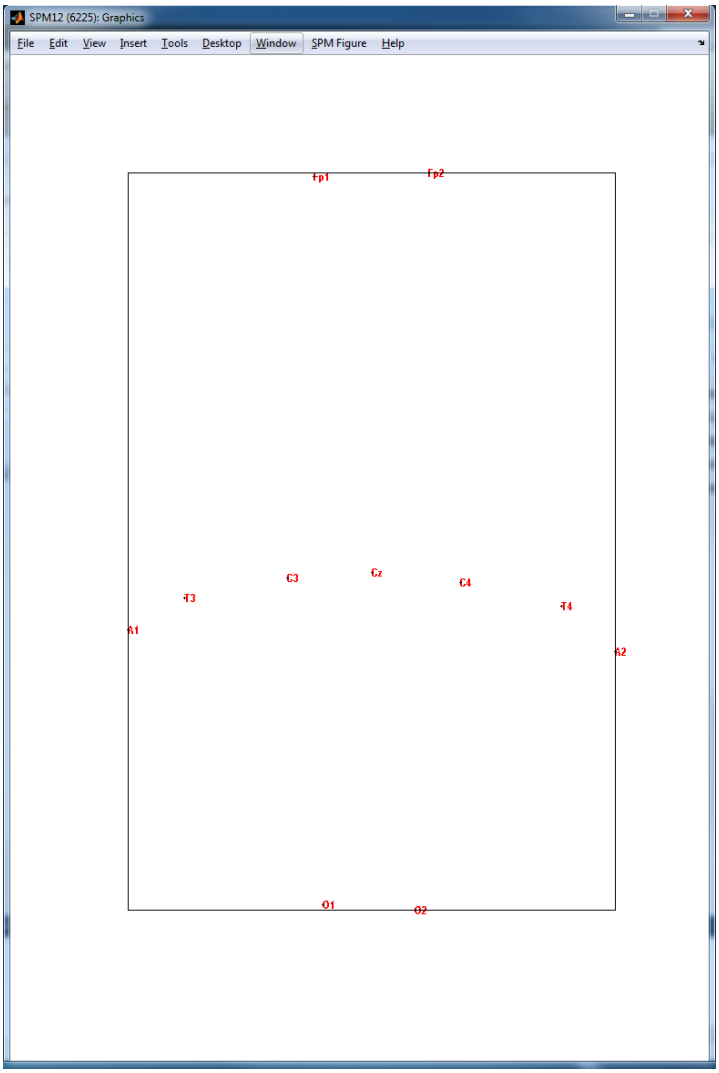


Figure 10: 2D Projection of electrode position set by default based on their names

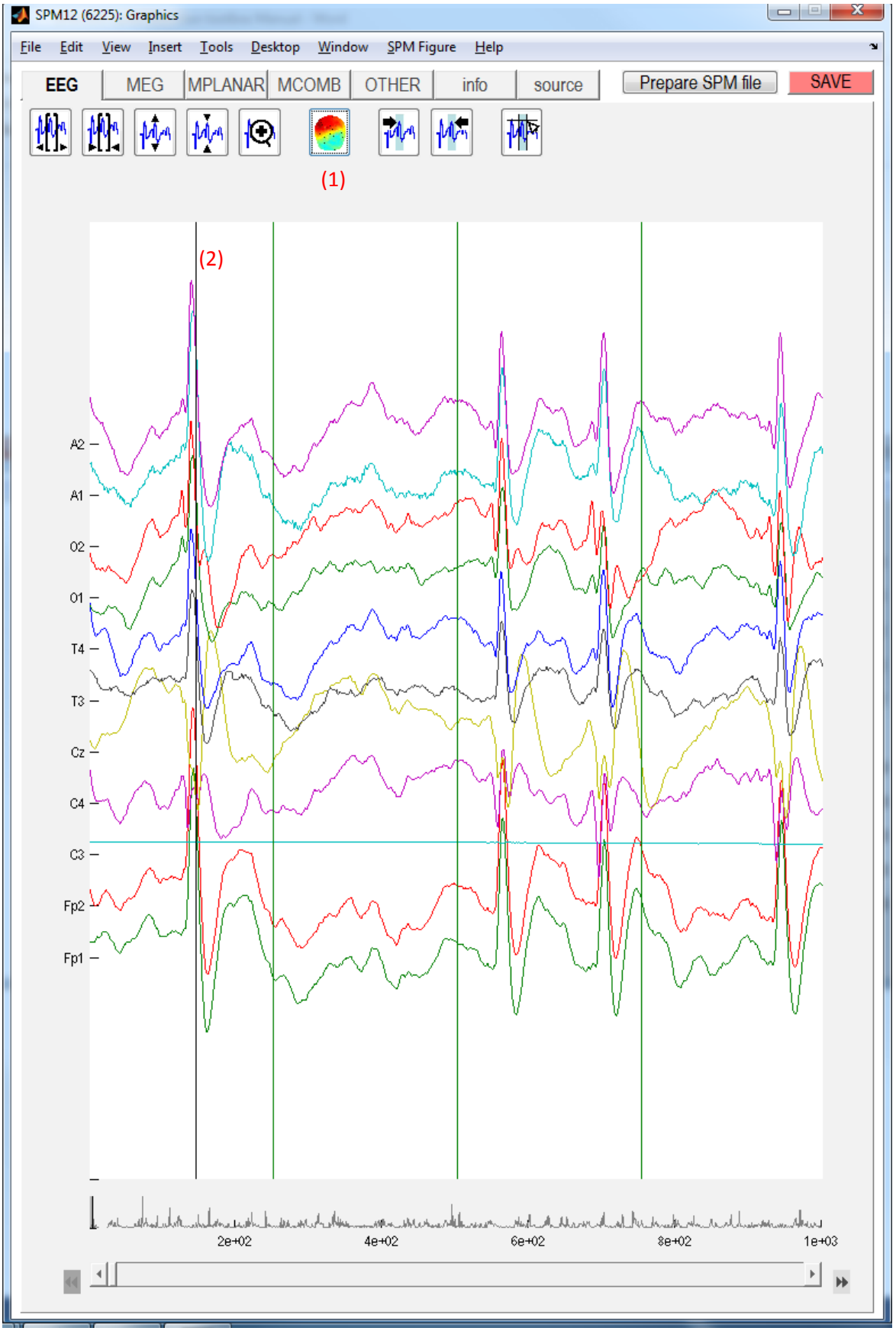
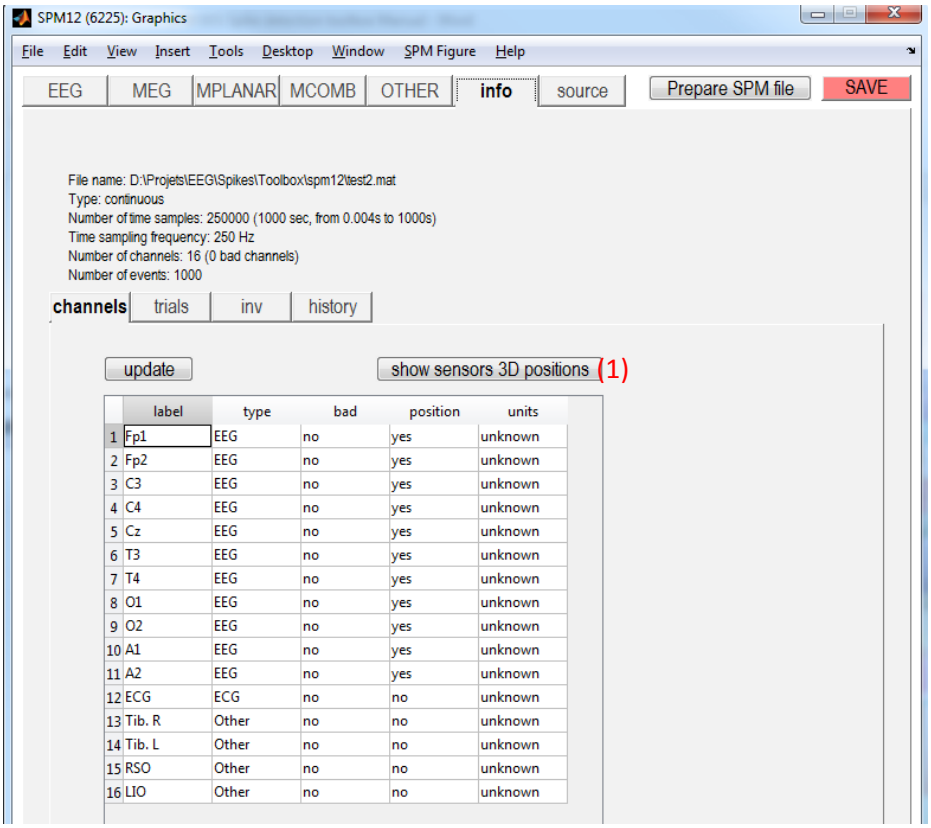


Figure 11: Graphics window after SPM file preparation



	label	type	bad	position	units
1	Fp1	EEG	no	yes	unknown
2	Fp2	EEG	no	yes	unknown
3	C3	EEG	no	yes	unknown
4	C4	EEG	no	yes	unknown
5	Cz	EEG	no	yes	unknown
6	T3	EEG	no	yes	unknown
7	T4	EEG	no	yes	unknown
8	O1	EEG	no	yes	unknown
9	O2	EEG	no	yes	unknown
10	A1	EEG	no	yes	unknown
11	A2	EEG	no	yes	unknown
12	ECG	ECG	no	no	unknown
13	Tib. R	Other	no	no	unknown
14	Tib. L	Other	no	no	unknown
15	RSO	Other	no	no	unknown
16	LIO	Other	no	no	unknown

Figure 12 : Tab. "info" in the Graphics window

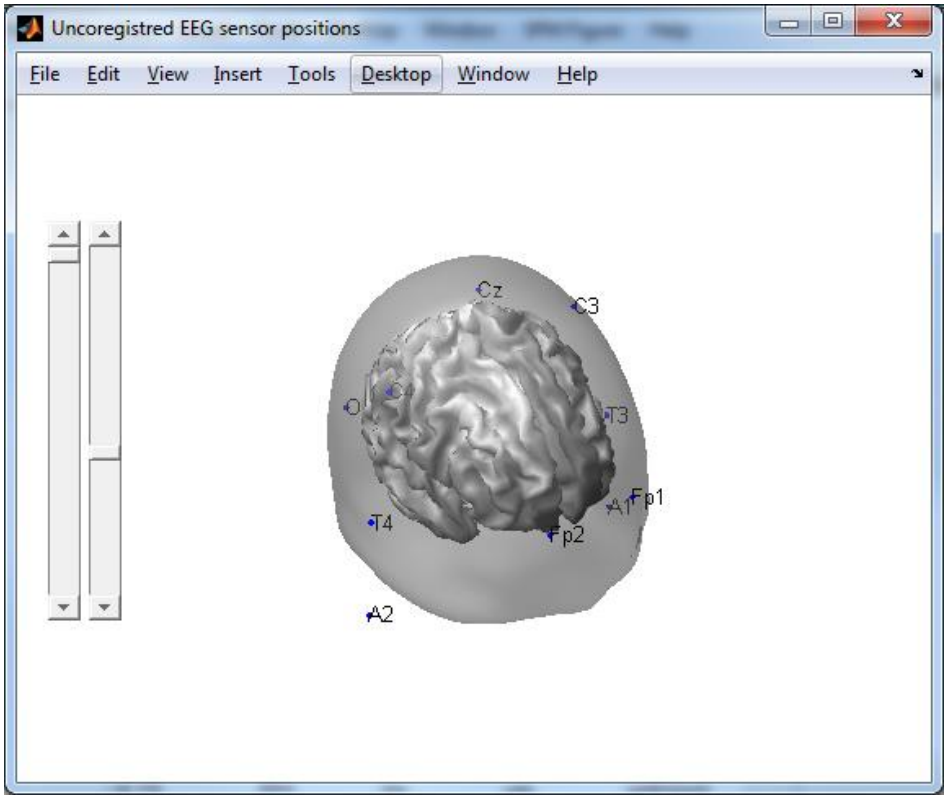


Figure 13 : EEG sensors position shown in 3D

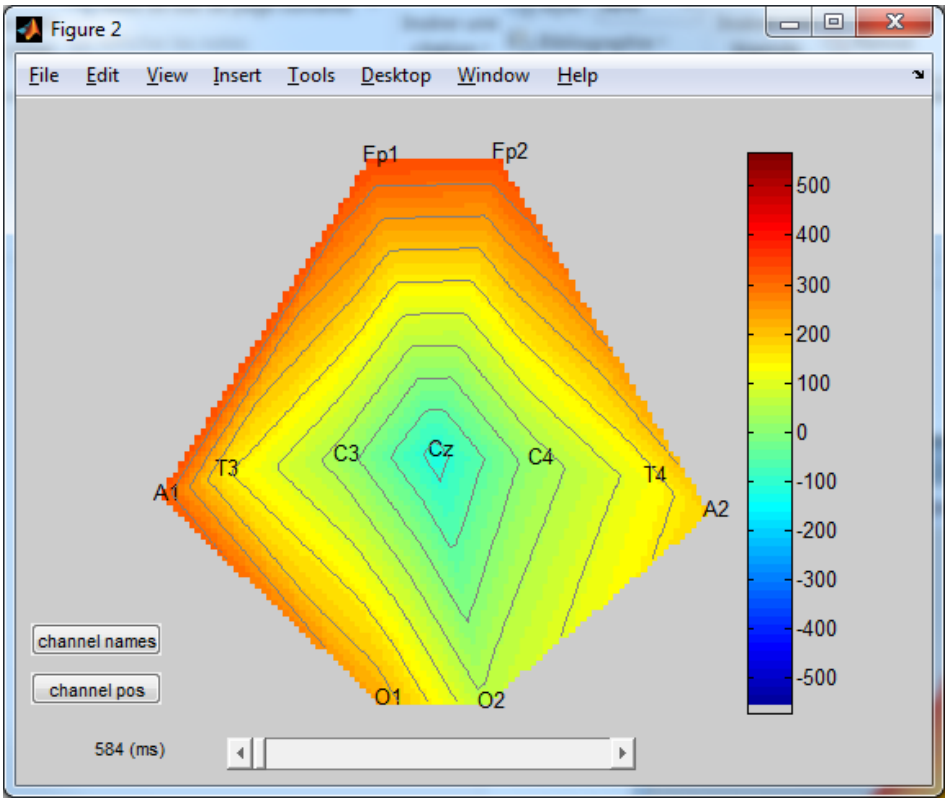


Figure 14 : Scalp interpolation near an epileptic spike

7 SPIKE DETECTION TOOLBOX

7.1 INTRODUCTION

For technical reasons, the spikes detection toolbox has been implemented in SPM12 as an "Artefact detection" module, even if obviously spikes have nothing to do with artefacts. The main reason behind this choice was the simplicity to integrate this kind of module in SPM. Moreover, the use of these "artefact" modules are straightforward because they always use the same guideline.

This section will mainly focus on explaining how to use this toolbox and how the algorithm detailed in [Nonclercq2012] and [Nonclercq2009] have been adapted to SPM.

7.2 GUIDELINE

The different steps to detect spikes in an EEG signal are the following:

1. In the SPM "Menu" window (Figure 1), select the "Detect artefacts" item in the "Preprocess" list box (2); The "Batch Editor" window for "Artefact detection" module shown in Figure 15 appear.
2. The "Batch Editor" window presents several items. You first have to select the SPM EEG file containing the spikes you want to detect by double-clicking on "File Name" (1); a "Select a M/EEG data File" window as shown in Figure 2 allows to do this. Then, you have to select the detection "Mode" (2) as "Mark" (not "Reject"). The "Spike detection" module only supports this mode. Finally, you have to specify the method (i.e. the module) by configuring "How to look for artefacts" (i.e. here spikes) (3). First, select "New: Method" in "How to look for

artefacts". Then, Select "Spikes" as "Detection algorithm" (see (4) and (3) in Figure 16). When a new method has been added, the options for this method (here, the spike detection algorithm called "Spikes") are displayed as shown in Figure 16. Change the option "Clear all events" (5) from "No" to "Yes" in order to remove all trials/events, this way clearing the boundary trials of the original EEG file. The remaining options will be detailed in the "Spike detection algorithm parameters" section (see below).

3. You now have to configure the channels on which the spike detection algorithm have to be executed (see (1) in Figure 16). There are several options:
    a. "All": select all the channels present in the file. This option is selected by default. Don't forget to delete it before specifying other options below (right click on "Channel selection" and delete item "All").
    b. "Select channels by type": you can select the channels based on their type, in our case, "EEG" should be specified (6). Please check that your file include EEG type! Remember that the type of an EEG channel can be set when preparing the file.
    c. "Custom channel" (2): directly "Specify" (6) the name of the channel as shown in Figure 16.
    d. "Regular expression": the channel name has to follow a given rule; for example, specify "A*" to select all channel names beginning with the letter "A".
4. You can run the detection algorithm by clicking on the green button play (7) of Figure 16.

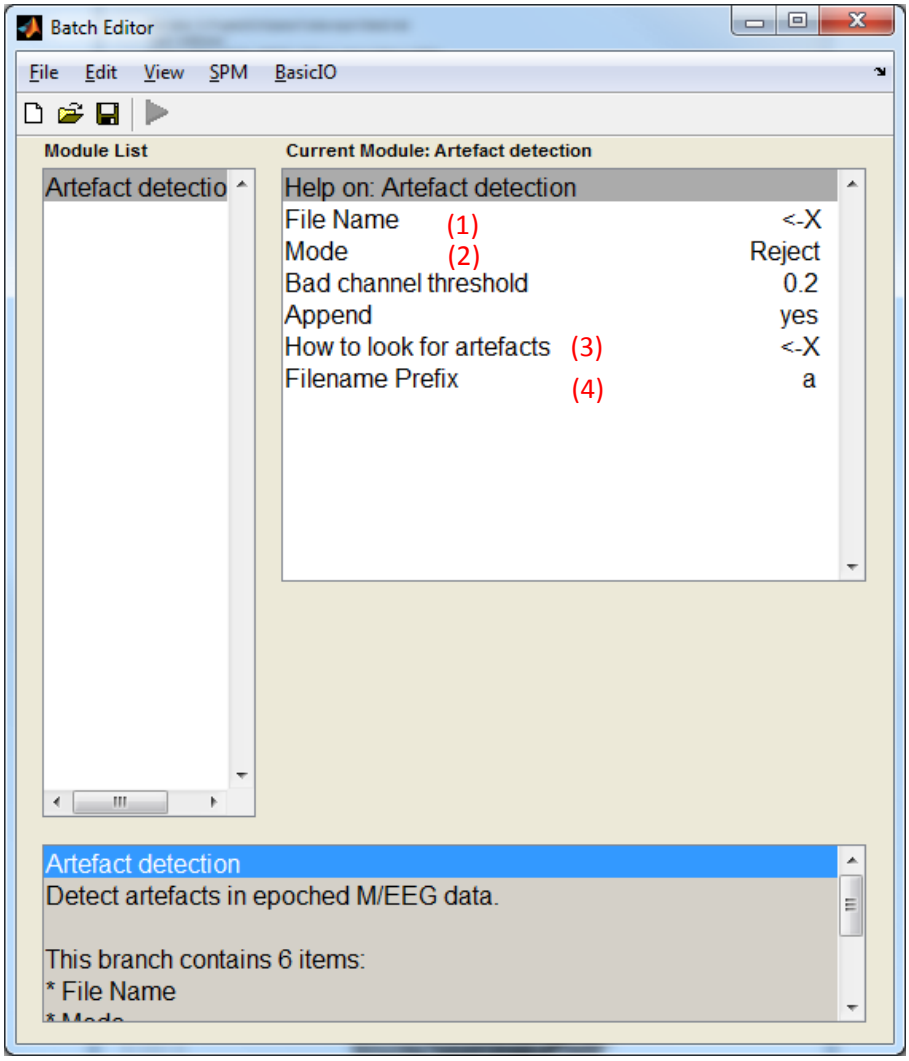


Figure 15: "Batch Editor" window for "Artefact detection" module

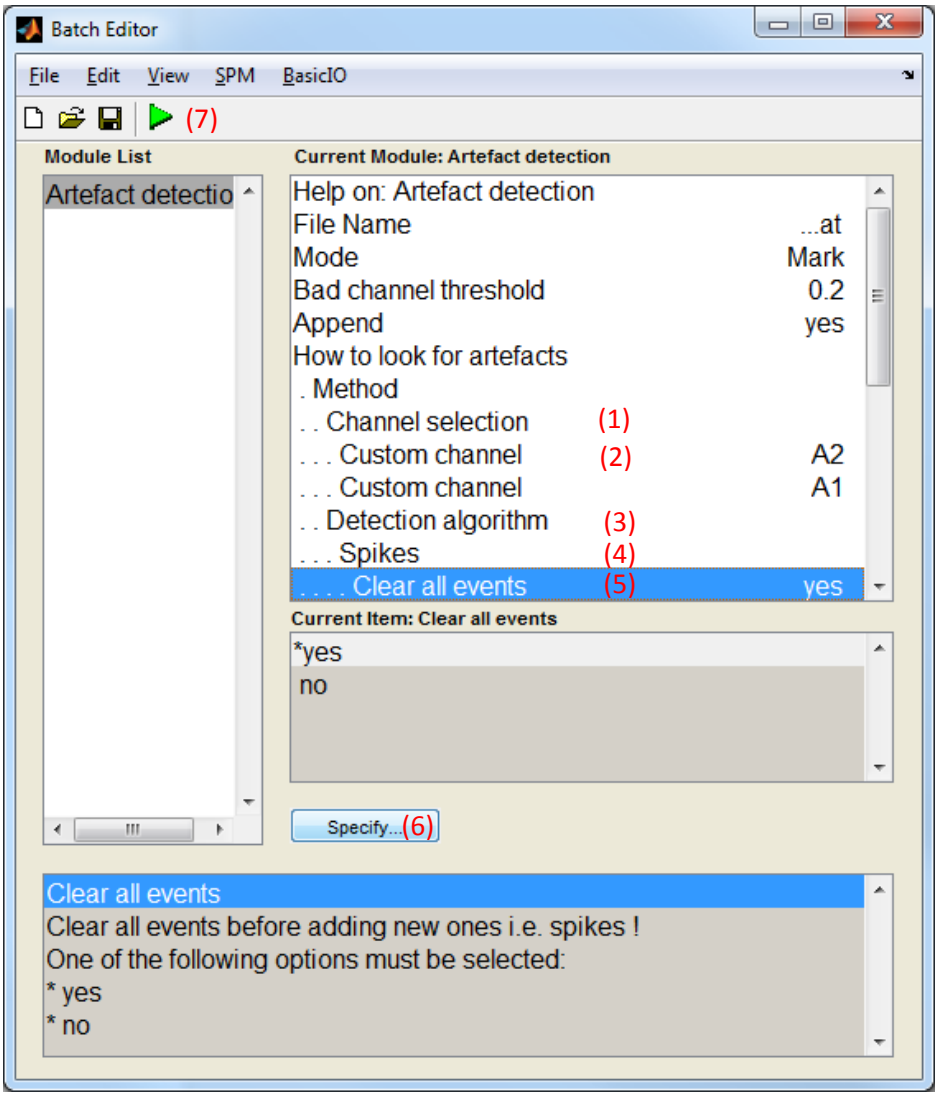


Figure 16: "Batch Editor" window for the "Spikes" sub-module of the "Artefact detection" module

7.3 RESULT OUTPUT

After running the spike detection algorithm, all detected spikes are added as events in a new SPM file. The new SPM filename is composed of a prefix added to the selected SPM filename; you can change this prefix by specifying it in "Filename prefix" item (4) of Figure 15. NB. It seems that SPM doesn't always take into account the "Filename prefix" change. To display the results, you have to select the "M/EEG" option in the "Display …" list box of the SPM menu (see Figure 1) and select the SPM header file (.mat), as explained in section "Display the SPM (EEG) file" (if it is not already done). If the file duration is very long or/and there are a lot of events in the file, display graphics can be very slow and Matlab can even crash (due to lack of memory). To avoid this problem, you can reduce the file length or/and reduce the number of channels by using the "Crop" module in the "Preprocess" window Menu (see (2) in Figure 1).

An example of detected spikes on electrodes A1 and A2 is presented in Figure 17 as vertical lines (respectively green and red). When clicking on an event (i.e. on a vertical line), you can obtain information about this event in the Matlab console, e.g. for a spike event: "Current event is selection

#4 /1279 (type= spike, value=A1).” where “type” is the event type i.e. spike and “value” gives the electrode name where this spike has been detected.

When running the spike detection algorithm, interesting information are given in the Matlab console. For each selected channel, the number of detected spikes and the spike index (SWI) are computed. Moreover, the total number of spikes that were detected trough all channels and the corresponding SWI are also computed. Another interesting data is the number of merged spikes where each detected spike that is timely near another spike is merged together. In this case, the corresponding SWI is also calculated. Let us notice that both Global SWI should always have close values.

Below an example of console output is shown when running the spike detection algorithm where you can find information described before e.g. the number of spikes detected in channel A1.

```
-----------------------------------------------------------------------
Running job #1
-----------------------------------------------------------------------
Running 'Artefact detection'

SPM12: spm_eeg_artefact (v6035)                16:53:05 - 20/04/2015
========================================================================

          SPM12: spm_eeg_artefact_spikes (v0001)     16:53:05 - 20/04/2015
          ------------------------------------------------------------
Clearing all events in file "atest2.mat" before adding spikes events!
Spike Detection for 2 channel(s) is started!
Number of spikes detected in channel A1 : 677 (SWI = 0.549)
Number of spikes detected in channel A2 : 602 (SWI = 0.498)
Number of spikes detected through all selected channels : 1279 (Global SWI = 0.549)
Number of merged spikes detected through all selected channels : 680 (Global SWI = 0.548)
There isn't a bad channel.
Done    'Artefact detection'
Done
```

A debug file is also created. Its filename is composed of a prefix “sp_” followed by the SPM filename that has been created during the detection. This debug file contains the selected channels data *D* in a Matlab structure *S*, the detection algorithm parameters called *DetectionParameters*, an array structure *sp* containing the index of each detected spikes and the corresponding SWI (*stat*) for each channel (the array index corresponds to the selected channel index) and a global data structure *GlobalSpikes* which contains the merged index of detected spikes and the global SWI.

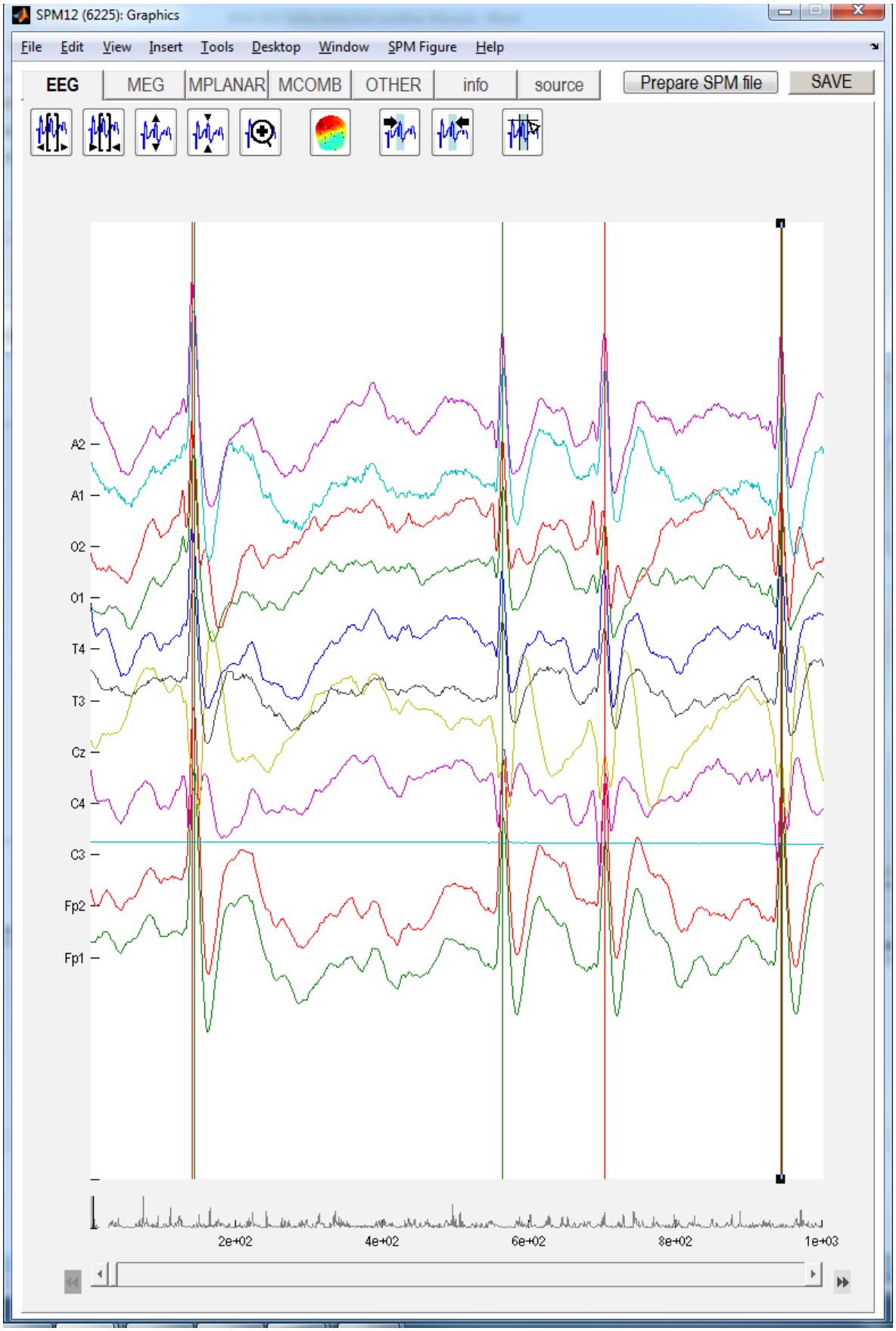


Figure 17: EEG Graphics view with detected spike events (two electrodes : A1 (green) & A2 ((red))

7.4 SPIKE DETECTION ALGORITHM PARAMETERS

This section will describe the different algorithm options, i.e. parameters which can be configured in the "Batch Editor" window shown in Figure 15 and Figure 16 for the spike detection algorithm. Figure 18 shows all the different parameters that can be tuned and their default values.

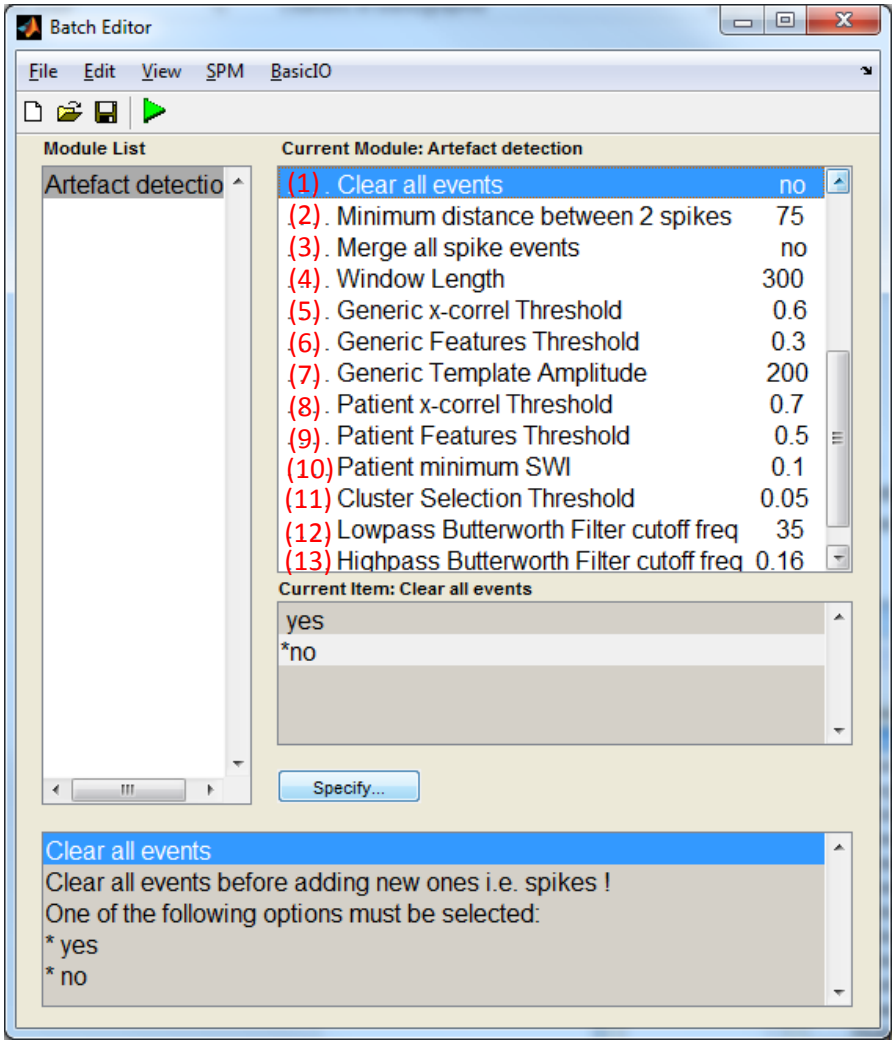


Figure 18: "Batch Editor" window with the spike detection algorithm parameters

Below, we give a short explanation of each parameter shown in Figure 18 for the spike detection algorithm. Some parameters will be more detailed in the next section related to the algorithm implementation. The numbers given between brackets in Figure 18 correspond to the numbers in this list:

1. Clear all events: remove all events of the original SPM file. Useful for removing the trial events of the opened file, that are present due to breaks/boundaries in the original EEG file (e.g. in EDF files).
2. Minimum distance between 2 spikes: minimum delay between two spikes (in ms).
3. Merge all spikes events: When this option is "Yes", it creates only one spike event for all selected channels only if the spike distance/delay between two spikes is less than the previous parameter (2).
4. Window Length: the duration/length (in ms) of the window containing the generic spike.

5. Generic x-correl Threshold: the normalized cross-correlation between the generic spike and a candidate spike has to be above this threshold to detect a spike.
6. Generic features Threshold: three features of the candidate spike are extracted: the rising slope, the falling slope and the curvature. Each features of a candidate spike must be greater than a fraction of the corresponding feature of the generic spike to detect a spike. This fraction is called Generic features Threshold.
7. Generic Template Amplitude: the amplitude of the generic spike given in µV.
8. Patient x-correl Threshold: the normalized cross-correlation between a patient specific spike template and the candidate spike has to be above this threshold to detect a spike.
9. Patient Features Threshold: The corresponding features threshold for a patient specific spike template.
10. Patient minimum SWI: When spikes are detected by the generic spike, a SWI is computed. The computed SWI has to be above this parameter to compute the patient specific spike detection.
11. Cluster Selection Threshold: the number of clusters used by the algorithm increases until one or more clusters contain less spikes (in %) than this threshold, typically less than 5% of the spikes.
12. Lowpass Butterworth Filter cutoff freq: Cut-off frequency of the zero phase low-pass Butterworth filter is applied to the EEG signal and to the generic spike. This filter is disabled when the input value is zero.
13. Highpass Butterworth Filter cutoff freq: Cut-off frequency of the zero phase high-pass Butterworth filter that is applied to the EEG signal and to the generic spike. This filter is disabled when the input value is zero.

7.5 SPIKE DETECTION IMPLEMENTATION

This section will describe the implementation of the spike detection algorithm for SPM, which was based on [Nonclercq2012] and [Nonclercq2009].

As explained before, the spike detection algorithm is included as an artefact detection module. For this purpose, the file/function called “spm_eeg_artefact_spikes” is added to the SPM folder. Several other files/functions called by this function are present in a “spikes” subfolder and all the filenames in this folder contain a prefix “sp_” for “spike module”.

The function “spm_eeg_artefact_spikes” therefore links SPM with the functions in the “spikes” subfolder implementing the spike detection algorithm. Its implementation deals with the different parameters shown in Figure 18. These can be configured in the Batch Editor for this artefact submodule i.e. a SPM configuration branch. Second, it adapts the SPM data structure for channels, user parameters and events to a simpler “per channel” data structure which can be directly used by the functions of the “spikes” subfolder. When all selected channels have been processed (a main “for” loop detects the spikes for each selected channel), the detected spikes for each channel are converted to events in the output file.

Before detecting spikes in “spm_eeg_artefact_spikes” function, the generic spike template is created by calling the “sp_generategenerictemplate” function. This function also computes the three features (curvature, rising and falling slopes) of this generic spike. The different steps to create this generic spike template are the following ones:

1. A 60ms stylized spike (triangle) is constructed at the beginning of a window. The total window length (in ms) and the stylized spike amplitude (in µV) are respectively given by the parameters (4) and (7) shown in Figure 18. Let us notice that the sampling frequency used to create this template is the same as the EEG sampling frequency.
2. Zero phase low-pass and high-pass Butterworth filters (5^{th} order) are applied to the stylized spike by using the function "sp_filter". The cut-off frequencies are respectively given by the parameters (12) and (13) shown in Figure 18. The respective filter is disabled when its cut-off frequency is zero.
3. A first derivation estimation is applied on the filtered stylized spike.
4. The derived signal is squared by keeping the sign of the original signal.
5. Finally, a smoothing is applied by averaging the squared signal (with sign) on a 50ms window.

Figure 19 shows the generic spike at the corresponding steps described above. The steps from 2 to 5 are implemented in the function "sp_preprocessing". This function is also applied to the EEG signal.

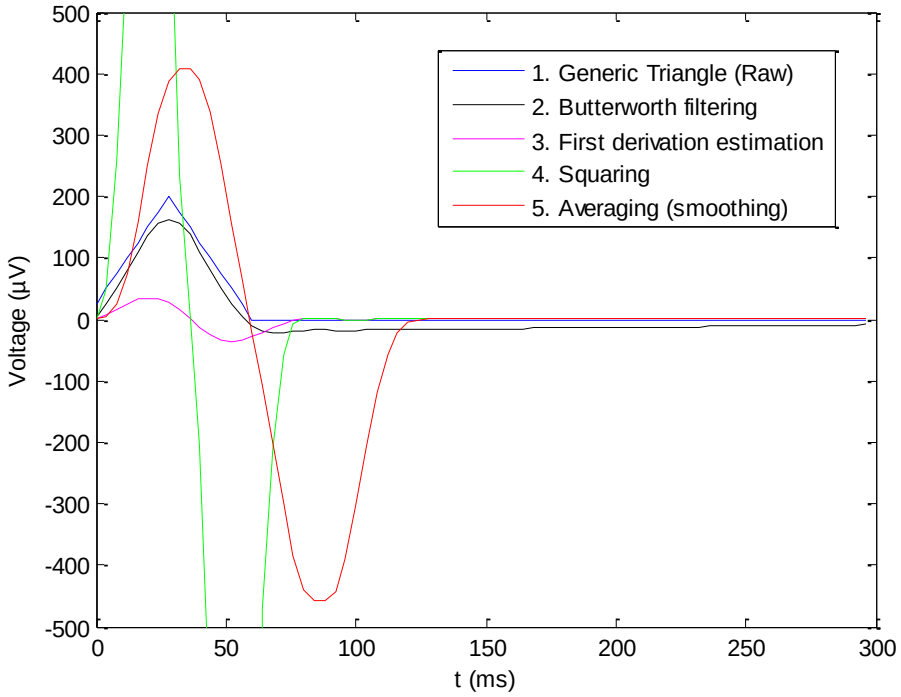


Figure 19: Different steps for generating the generic spike template

The different steps to detect spikes in each selected channel are executed in the main "for" loop of the function "spm_eeg_artefact_spikes":

1. The current EEG channel samples are extracted in an array called "RawData".
2. The current channel "RawData" is pre-processed to highlight the spike characteristics with respect to the background. The corresponding function is called "sp_preprocessing" and the output result is stored in the "ProcessedData" array. The pre-processing steps performed this function are identical to the ones described above for the spike template creation (steps 2 to 5). An illustration of pre-processing signal (from channel A2 of Figure 17) is shown in Figure 20.
3. The spike detection algorithm is then applied on the current pre-processed EEG channel by using the generic spike template. The corresponding function is called "sp_spikedetection" :
    a. The normalized cross-correlation between the signal and the template is computed by calling the Matlab function "normxcorr2" of the Matlab "Image Processing Tooblox".

   b. All maximum correlation values that are above the threshold (given by parameter (5) in Figure 18) that are separated by a minimum distance (parameter (2) shown in Figure 18) are considered as potential spikes. These values are found by using the Matlab function “findpeaks” of the Matlab “Signal Processing Toolbox”.
   c. For each potential spike, the three features are computed in a window whose length is equal to the one of the template. The beginning of this window is located at the maximum correlation value, corresponding to the beginning of the (pre-processed) spike.
   d. If the three features are above a fraction (parameter (6) shown in Figure 18) of the template features, the spike is validated.
   e. The position of the maximum of each validated spike is computed and then stored as an event. The “C matlab mex” function “sp_adj_spikes_pos” looks for this maximum (in RawData) around the beginning of the spike (step d) by using the parameter (2) of Figure 18 as search limits (before and after the current position). Figure 21 shows an illustration of detected spike, showing for each spike beginning and maximum positions.
4. Once the generic spike template detection has been performed, the SWI is computed with the function “sp_stats”. The SWI was obtained by dividing the number of seconds presenting one or more spikes by the length of the extract (in %). The algorithm divides the extract in 1 sec windows and checks if there is one or more spikes in these windows.
5. If the SWI computed during the first detection is above a given threshold ((11) in Figure 18), a second detection is performed using patient specific templates. The patient specific templates are computed by clustering as explained in [Nonclercq2012] and as implemented in the function “sp_clustersfromdetect”:
   a. Spikes are characterized by their pre-processed time series (i.e. the input of the clustering algorithm is made of the time series of each detected spikes that underwent the pre-processing defined in the previous section).
   b. Starting from one single cluster, the number of clusters increases (“NumClusters” variable) until one or more clusters contain less spikes than a given threshold ((11) in Figure 18, typically 5%).
   c. The energy is used to sort spikes and give a starting point for clustering. Sorted spikes are initially divided in “NumClusters” classes in order to generate a starting centroid (seed for the “kmean” function). After that, “kmean” clustering methods uses a distance based on correlation (by minimizing one minus the correlation).
   d. Finally, all clusters having less than the threshold (11) shown in Figure 18 are removed. The centroids of all the other clusters constitutes the patient’s specific spike templates.
6. After clustering has been processed, the “sp_detfromclusters” function uses each the corresponding centroids to detect spikes with the “sp_spikedetection” function. For each detection, the centroid is used as a patient specific template and the three features are adapted (averaging the features of detected spikes belonging to the current cluster). Since a given spike can be detected more than once (as a single spike can be detected by various centroids, doublets may appear), the “C matlab mex” function “sp_merge_epoch” merges detected spikes together when the distance between them is less than the parameter (2) shown in Figure 18.
7. After the patient specific detection is finished, if the parameter (3) shown in Figure 18 is set to “no”, spikes are directly added as events by using the “add_events” sub-function. The type of this event is set to “spike” and its value is set to the current channel name. Let us notice that,

if the parameter (3) shown in Figure 18 is set to “yes”, spikes of all channels are merged together by using the “sp_merge_epoch” function after the execution of the “for” main loop.

8. Finally, the SWI and the number of detected spikes in the current EEG channel are displayed in the Matlab console (using the “sp_stats” function).

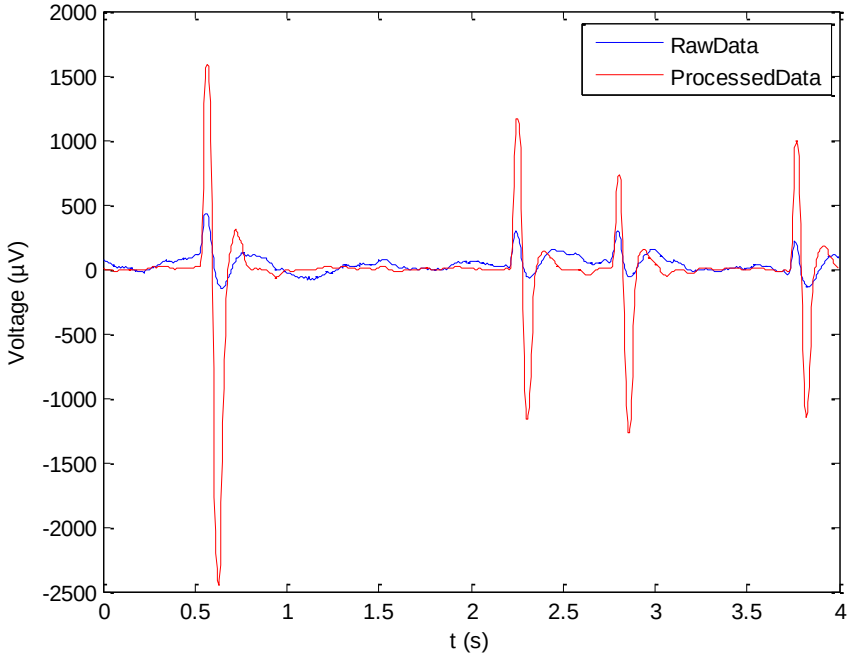


Figure 20: Signal example after and before signal pre-processing

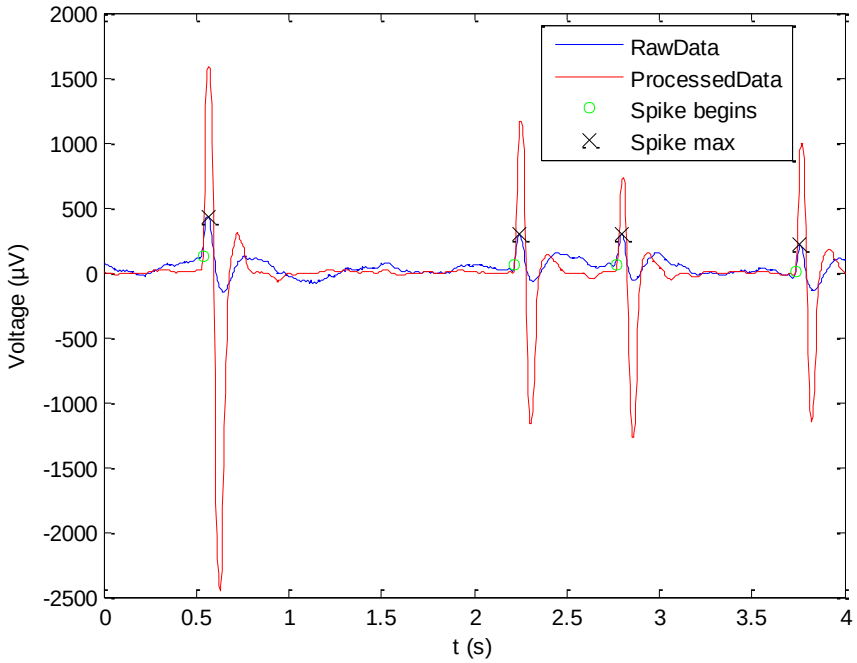


Figure 21: Spike position example

Let us notice that some algorithm details were not explained here but should be easily understood reading the Matlab code.

8 ACKNOWLEDGEMENT

The developers of this SPM toolbox, Antoine Nonclercq & Rudy Ercek, would like to thank all [Nonclercq2009] and [Nonclercq2012]'s co-authors, namely Martine Foulon, Denis Verheulpen, Cathy De Cock, Marga Buzatu, Pierre Mathys and Patrick Van Bogaert who helped to develop the original method implemented in this toolbox.

They also want to thank Professors Philippe Peigneux and Xavier De Tiège (ULB) for testing this toolbox and validating the method.

Finally, they also thank the SPM team for developing this toolbox, more particularly Vladimir Litvak and Guillaume Flandin (UCL) for their advices in the development process.

9 BIBLIOGRAPHY

[Ashburner2013] Ashburner, J., Barnes, G., Chen, C., Daunizeau, J., Friston, K., Kiebel, S., … Stephan, K. (2013). SPM12 Manual The FIL Methods Group (and honorary members). *Functional Imaging Laboratory*, 475–1. doi:10.1111/j.1365-294X.2006.02813.x

[Nonclercq2009] Nonclercq, A., Foulon, M., Verheulpen, D., De Cock, C., Buzatu, M., Mathys, P., & Van Bogaert, P. (2009). Spike detection algorithm automatically adapted to individual patients applied to spike and wave percentage quantification. *Neurophysiologie Clinique*, *39*, 123–131. doi:10.1016/j.neucli.2008.12.001

[Nonclercq2012] Nonclercq, A., Foulon, M., Verheulpen, D., De Cock, C., Buzatu, M., Mathys, P., & Van Bogaert, P. (2012). Cluster-based spike detection algorithm adapts to interpatient and intrapatient variation in spike morphology. *Journal of Neuroscience Methods*, *210*(2), 259–265. doi:10.1016/j.jneumeth.2012.07.015