Technical Document

# DBSFILT v0.18b

---

An open-source deep brain stimulation-induced artifacts filtering toolbox


Guillaume Lio [1,2,3]
Stéphane Thobois [1,2,3,4]
Bénédicte Ballanger [1,2,5]
Brian Lau [6]
Philippe Boulinguez [1,2,5]*
*[1] Université de Lyon, F-69622, Lyon, France*
*[2] Université Lyon 1, Villeurbanne, France*
*[3] CNRS, Centre de Neuroscience Cognitive, Bron, France*
*[4] Hospices civils de Lyon, hôpital neurologique Pierre Wertheimer, Bron, France*
*[5] INSERM U1028, CNRS UMR5292, Centre de Recherche en Neurosciences de Lyon, Lyon, France*
*[6] Sorbonne Universités, UPMC Univ Paris 06, UMR S 1127, CNRS UMR 7225, Institut du Cerveau et de la Moelle épinière, F-75013, Paris, France*

# *Introduction*

DBSFILT is a matlab toolbox designed to detect and remove deep brain stimulation induced artifacts from EEG or MEG data.

This tutorial demonstrates how to use DBSFILT, and provides hands-on learning experience by operating on the tutorial EEG dataset :

*DBSFILT_Example_Files/ DBSFILT_P1_dbs_ON_EC.set*

With this dataset, you should be able to reproduce the sample actions discussed in the tutorial.

# *1- Installation*

Requirements for DBSFILT:

- MATLAB R2008b or above.
- The MATLAB Signal processing toolbox.
- EEGLAB for .set files processing.

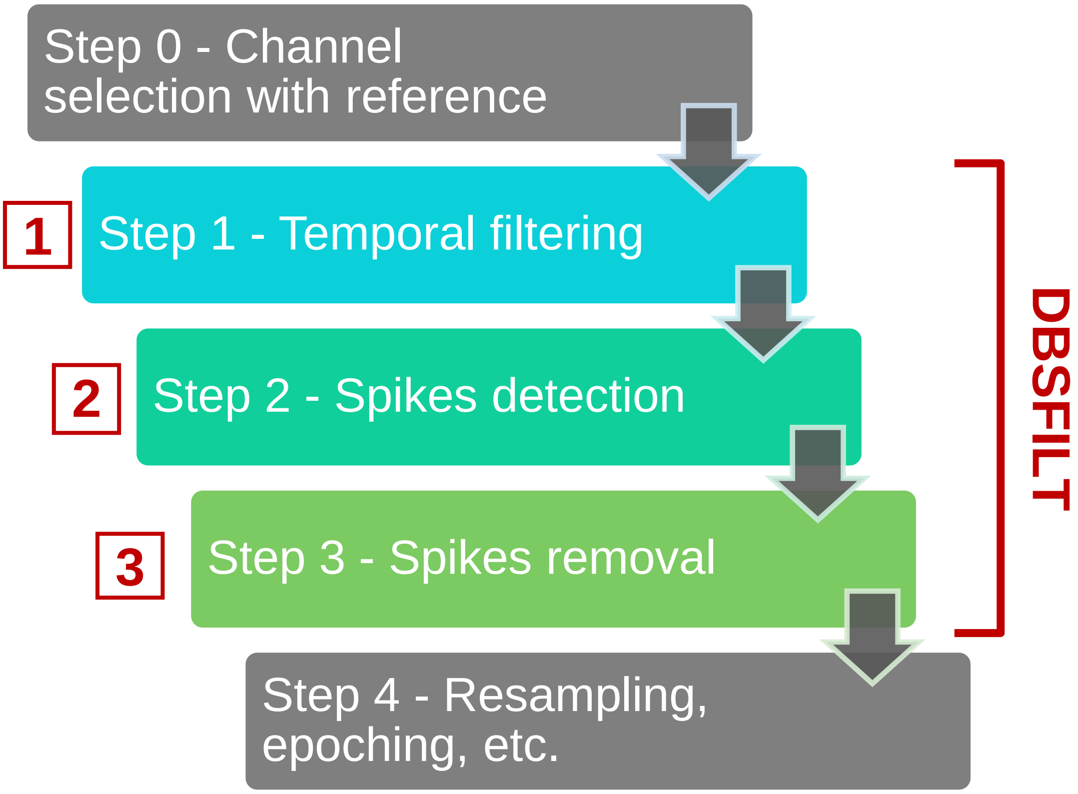
  *https://sccn.ucsd.edu/eeglab/index.php*

1) Download the DBSFILT package from:
   http://sourceforge.net/projects/dbsfilt/files/
   *Download the most recent .zip file (DBSFILT_V_0.18b.zip as of the writing of this document).*

2) Unzip the package, and add the directory and all subdirectories to your MATLAB path.

3) Run “DBSFILT” from the MATLAB command line.

## *2- DBSFILT pipeline*

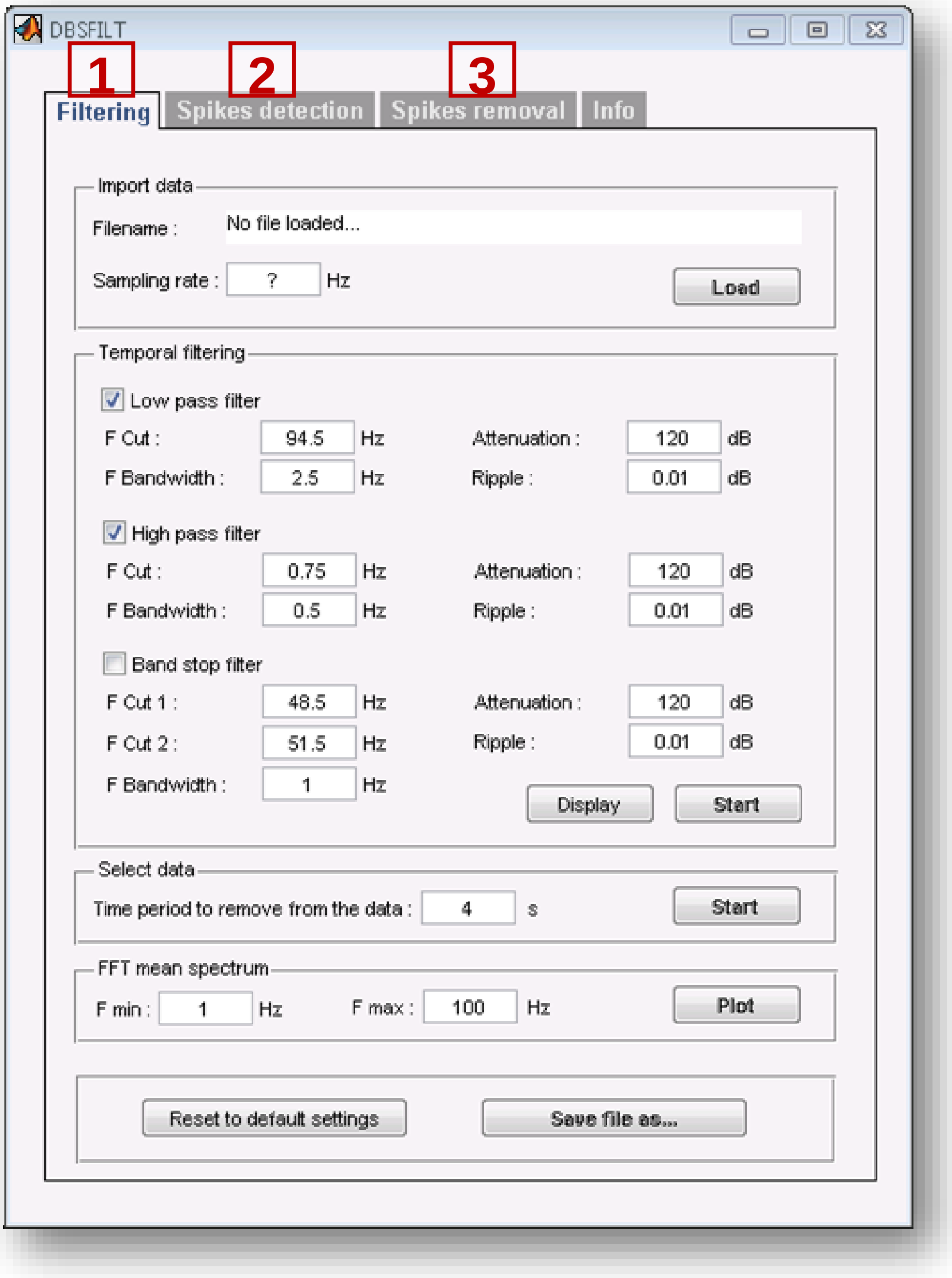


# 3- Graphical user interface

The DBSFILT graphical user interface (GUI) is designed to allow non-experienced Matlab users to apply filtering techniques to their data.

Launch the GUI by running 'DBSFILT' from the matlab command line (>> DBSFILT). The GUI is divided into 3 main panels, one panel for each processing step:

- Step 1 – Temporal filtering [1]
- Step 2 – Spikes detection [2]
- Step 3 – Spikes removal [3]



# *4- DBS artifacts filtering:*

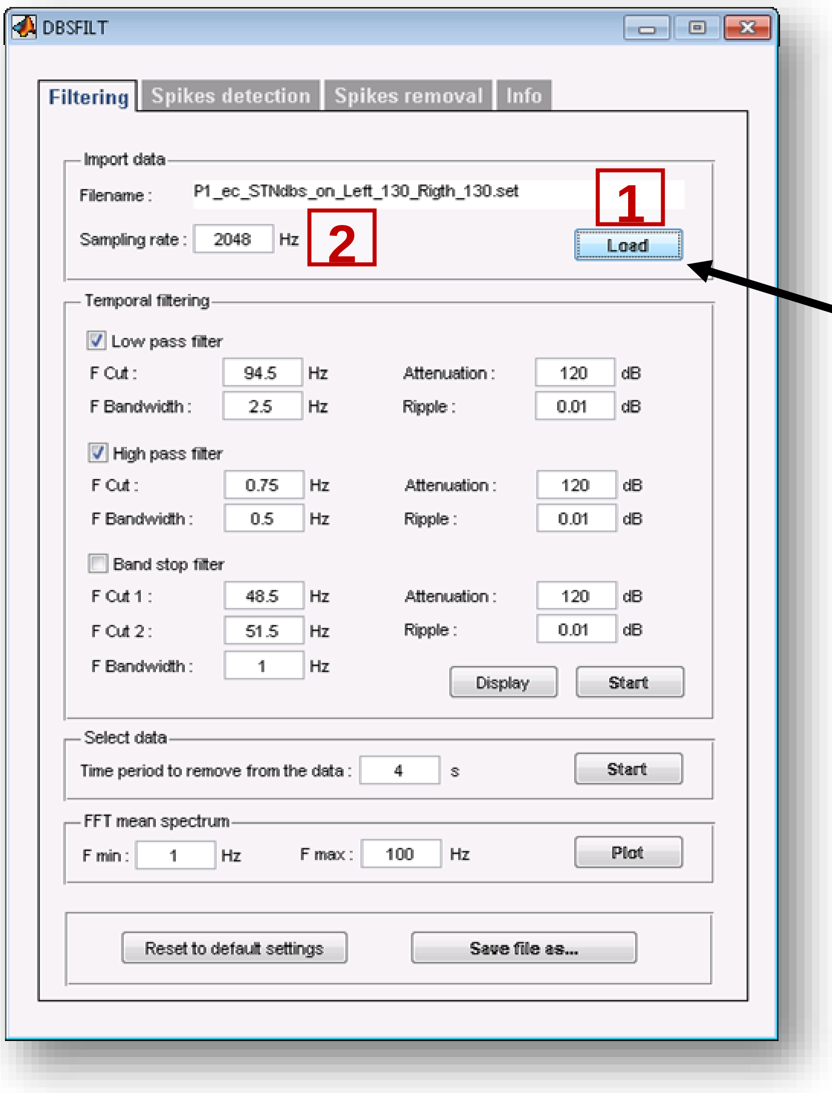
## *– Step 1, temporal filtering.*

***Step 1.1*** – Load the EEG data file [1]. Two types of files are supported:

- .set EEGLAB files, **<u>not epoched</u>** and restricted to EEG electrodes.
- .mat MATLAB files (a.b) with a=number of channels, b=number of samples.

If a .set file is selected, the sampling rate is detected automatically.

If a .mat file is selected, the sampling rate has to be set manually [2].




**Step 1.1** – Load the .set file (e.g. *DBSFILT_P1_dbs_ON_EC.set)*.

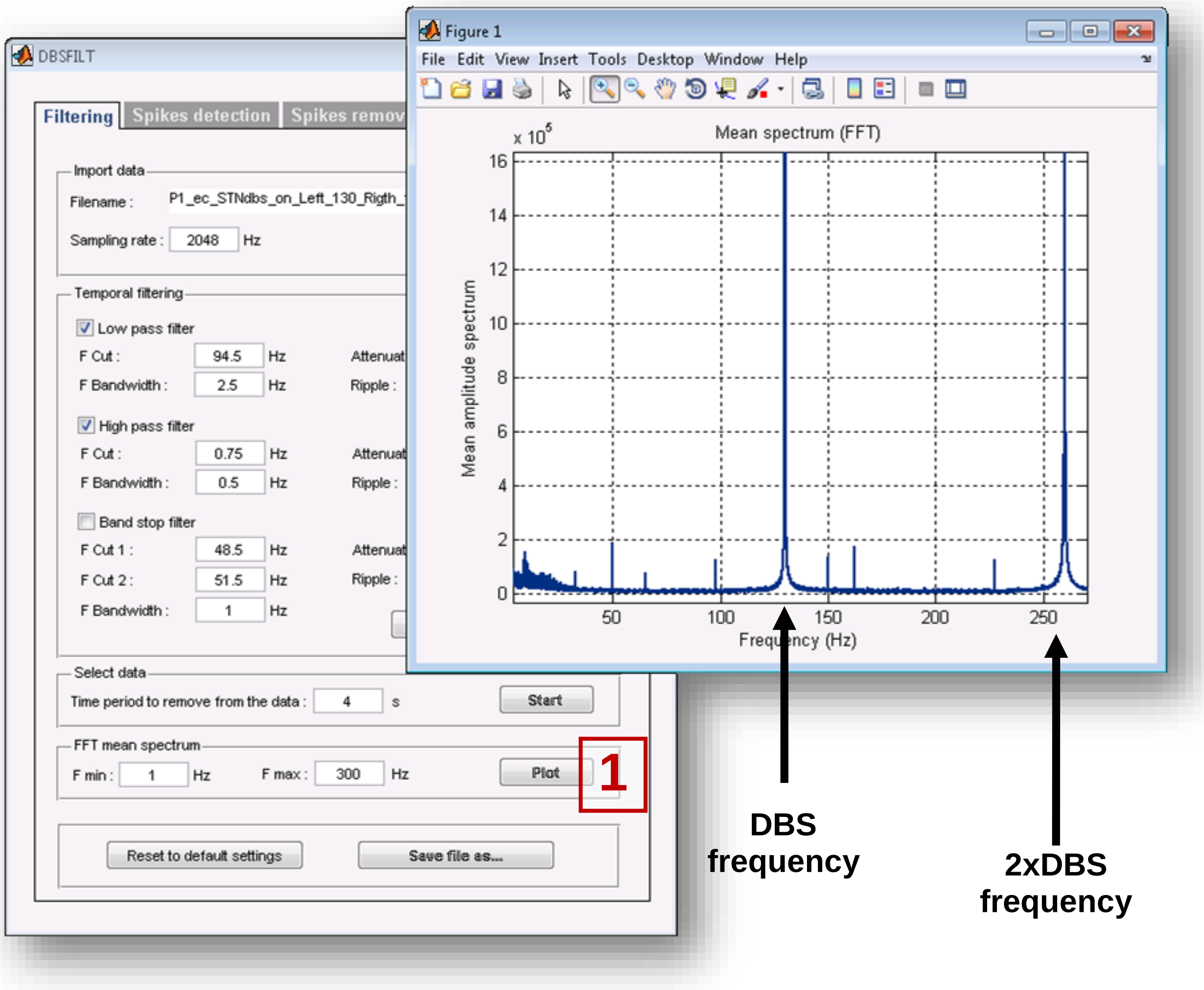
The sampling rate is set automatically.

***Step 1.2*** –

To plot the Fast Fourier Transform (FFT) amplitude spectrum, averaged on all channels, select “Plot” in the “FFT mean spectrum” box [1]. In order to visualize DBS artifacts, you should set “F Max” to at least 300 Hz.

At each step of the DBS filtering, the “FFT mean spectrum” box can be used to monitor the process.

*Tutorial EEG dataset: Note the huge spikes at Deep Brain Stimulation Frequency (130 Hz) and its multiples (e.g., 260 Hz).*

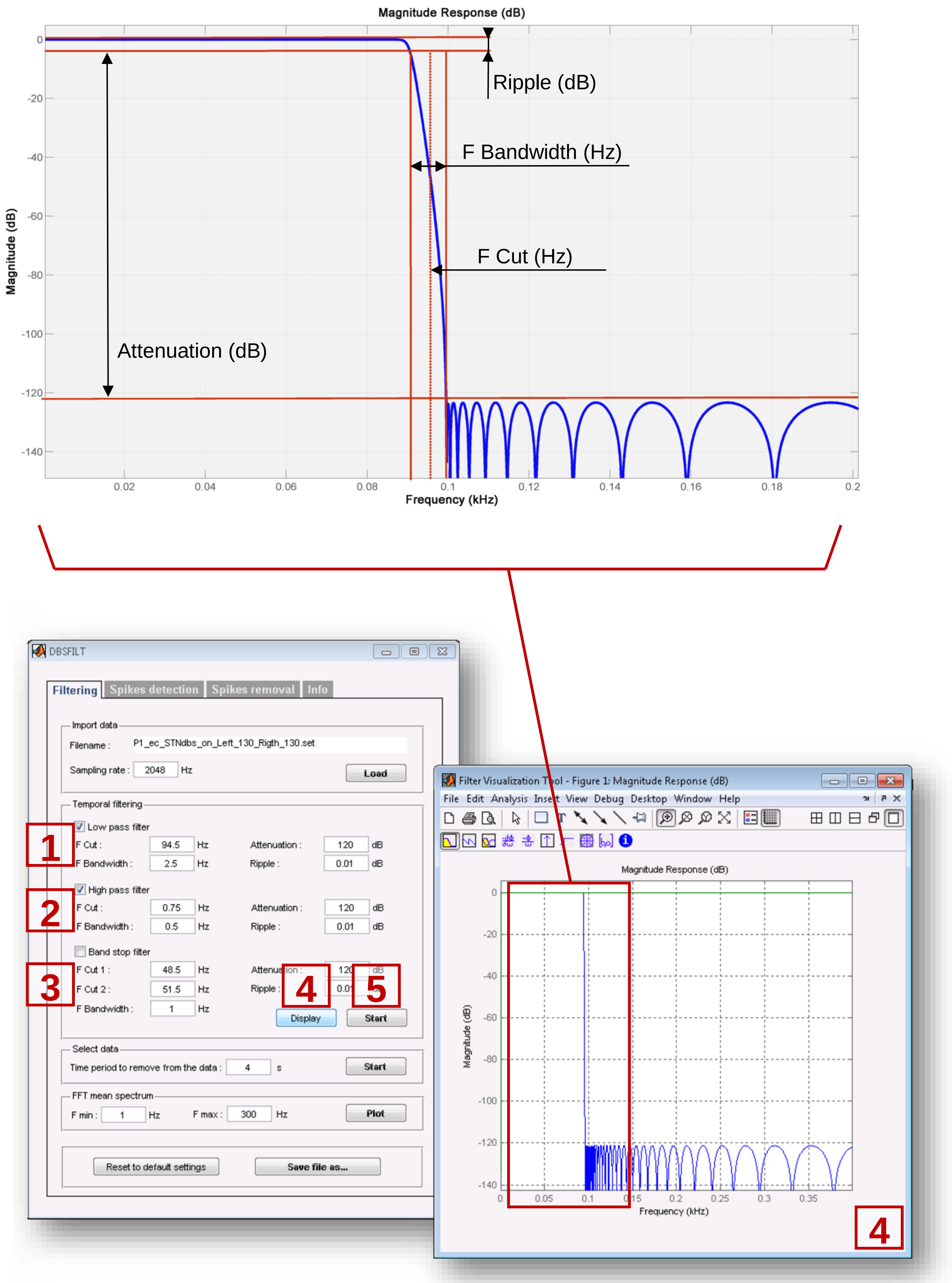


***Step 1.3*** – Temporal filtering. A large part of DBS induced activity can be filtered by combining a Low pass and a High pass filters to keep the EEG activity in a frequency band below the stimulation frequency.

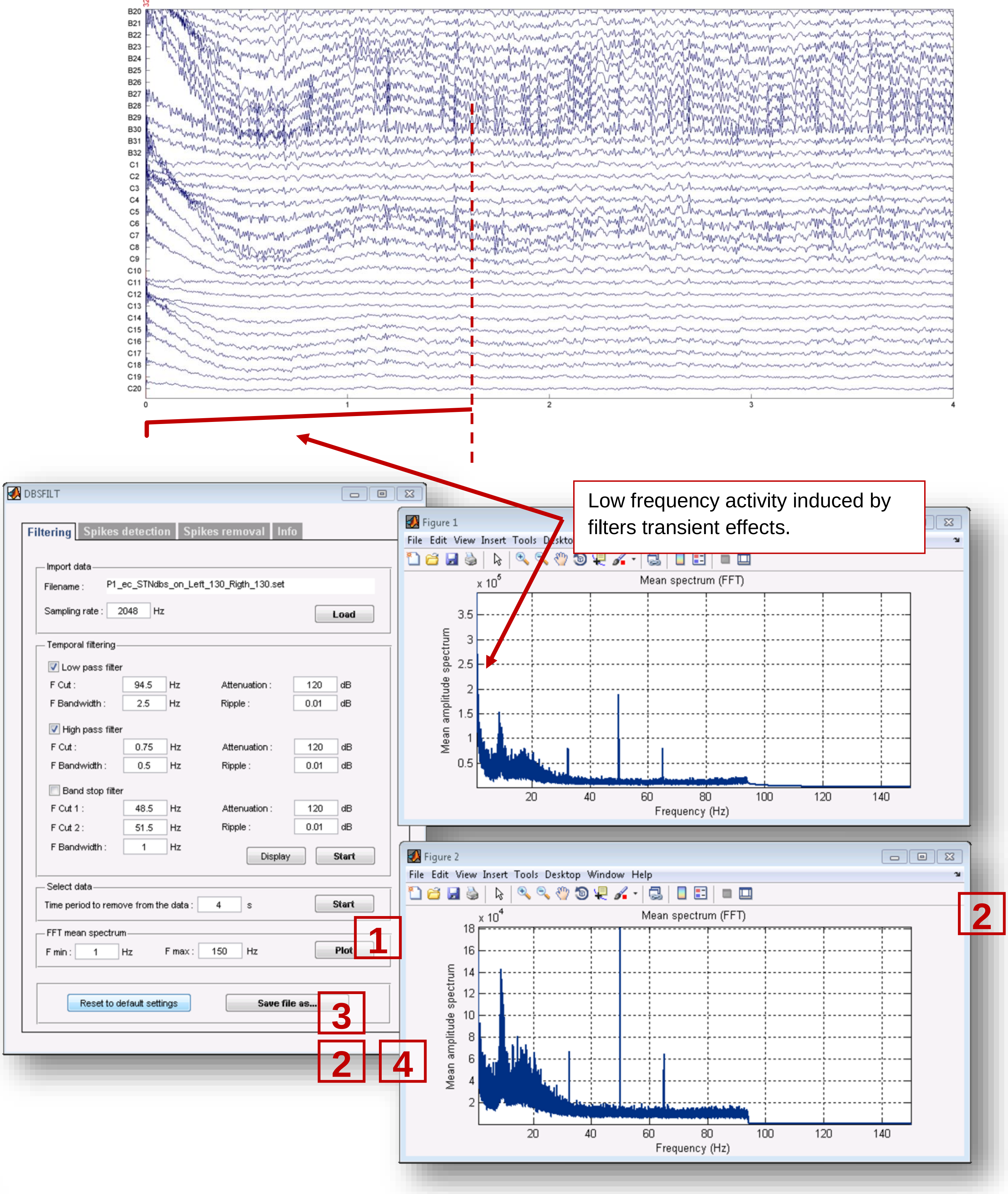
DBSFILT allows setting user defined parameters in the "Temporal filtering" panel (F cut, F Bandwidth, Ripple, Attenuation) to design a IIR Chebishev Type II filters of minimum order that will be applied to the data. Low pass [1], high pass [2] and band stop [3] filters characteristics can be plotted with the 'display' button [4]. Filtering is launched with the start button [5].

Note that the IIR Chebishev Type II filters are applied in both forward and reverse directions (see Matlab filtfilt and cheby2 functions). They are zero-phase filters, monotonic in the passband (no ripple / constant magnitude response) and equiripple in the stopband.

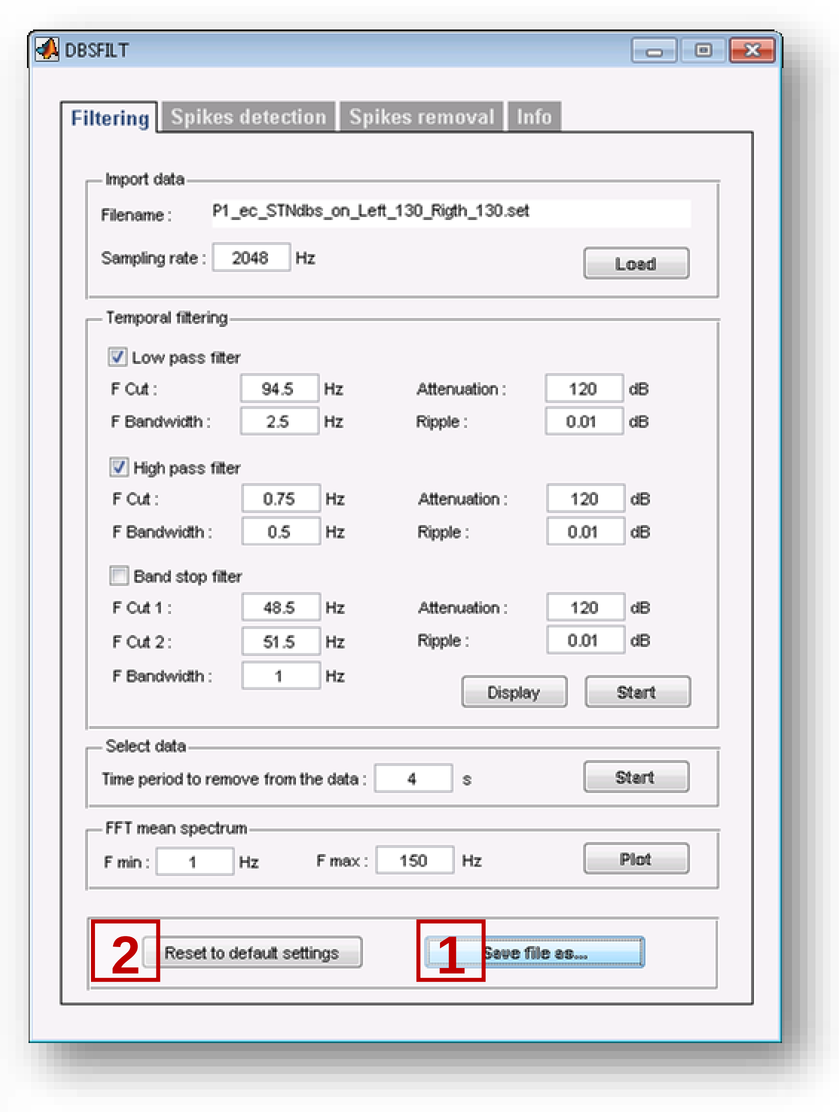
## *Example of a low-pass and high-pass filter design:*



***Step 1.4*** – To suppress the transient effects of the filters at the beginning and at the end of the time series, the first and last seconds where edge effects occur must be removed [3]. This can be done by means of the ‘select data’ box.



***Step 1.5*** – . Save the filtered data [1] and clear all data in memory with the ‘reset’ button [2] if necessary.



## *5- DBS artifacts filtering:*

### *– Step 2, spikes detection.*

While pass-band filtering removes most of DBS induced artifacts, remaining aliased DBS frequencies must still be removed. DBSFILT uses the following properties of the DBS artifacts to facilitate identification:

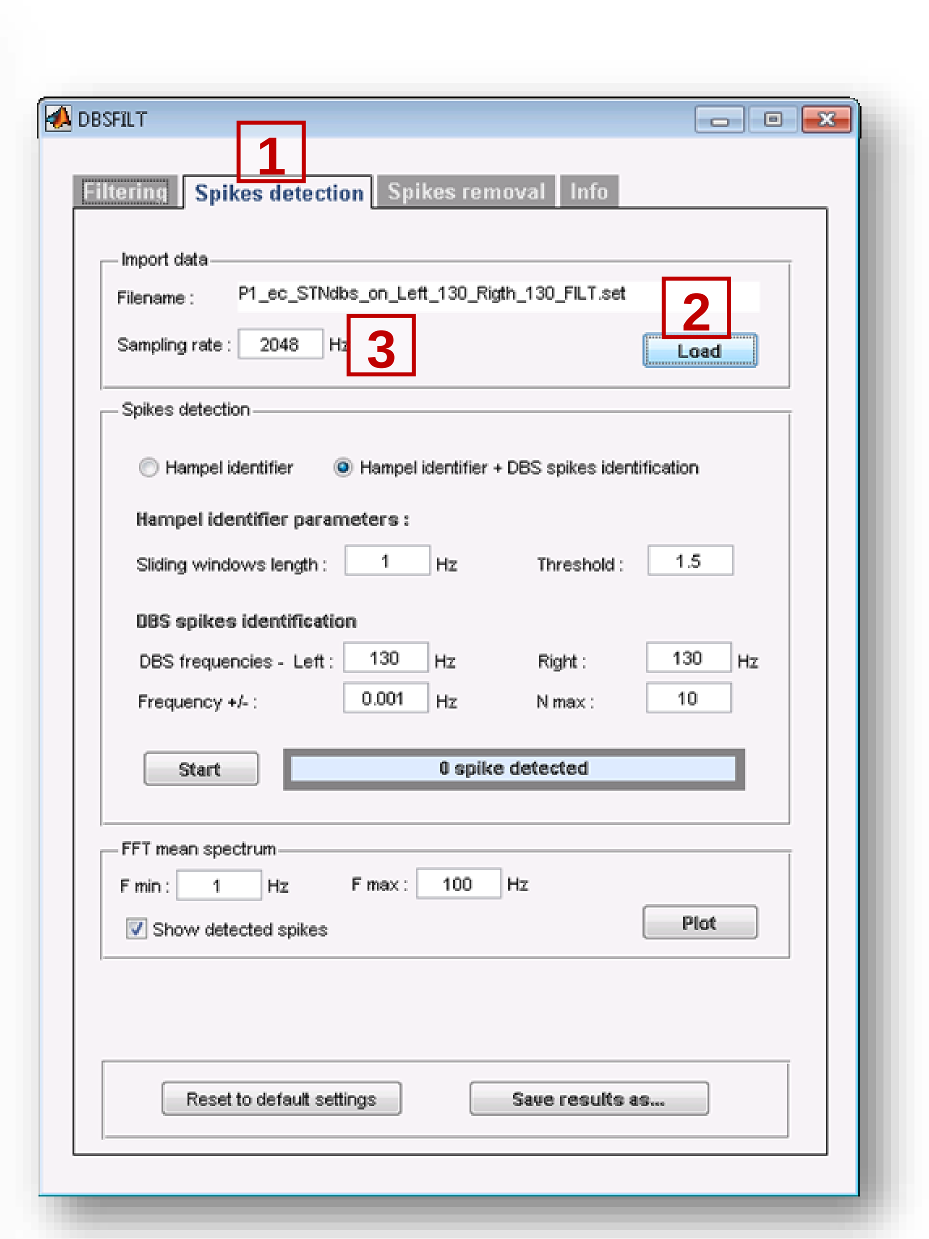
- DBS induced artifacts are spread on all electrodes.
- DBS induced artifacts are stationary.
- DBS induced artifacts are aliased frequencies of the fundamental and harmonic frequencies of the stimulation frequency.
- DBS induced artifacts power is higher than the power of the background neuronal activity in the same frequency window.

These properties make DBS induced artifacts detectable in the frequency space as spikes in the (electrodes averaged) amplitude spectrum.

Two methods can be used to identify these spikes:

- A frequency domain Hampel filtering (Allen et al. 2010). This is detailed below in ***Step 2.2a***
- A frequency domain Hampel filtering using a supplementary routine to guide the filter towards the frequencies at which DBS artifacts are likely to occur. This is detailed below in ***Step 2.2b***.

***Step 2.1*** – Select the 'Spikes detection' tab [1], load the file issued from the first step [2] and, if necessary, add the sampling rate [3].



**_Step 2.2a_** – . Spikes detection with the standard Hampel identifier (Allen et al. 2010):

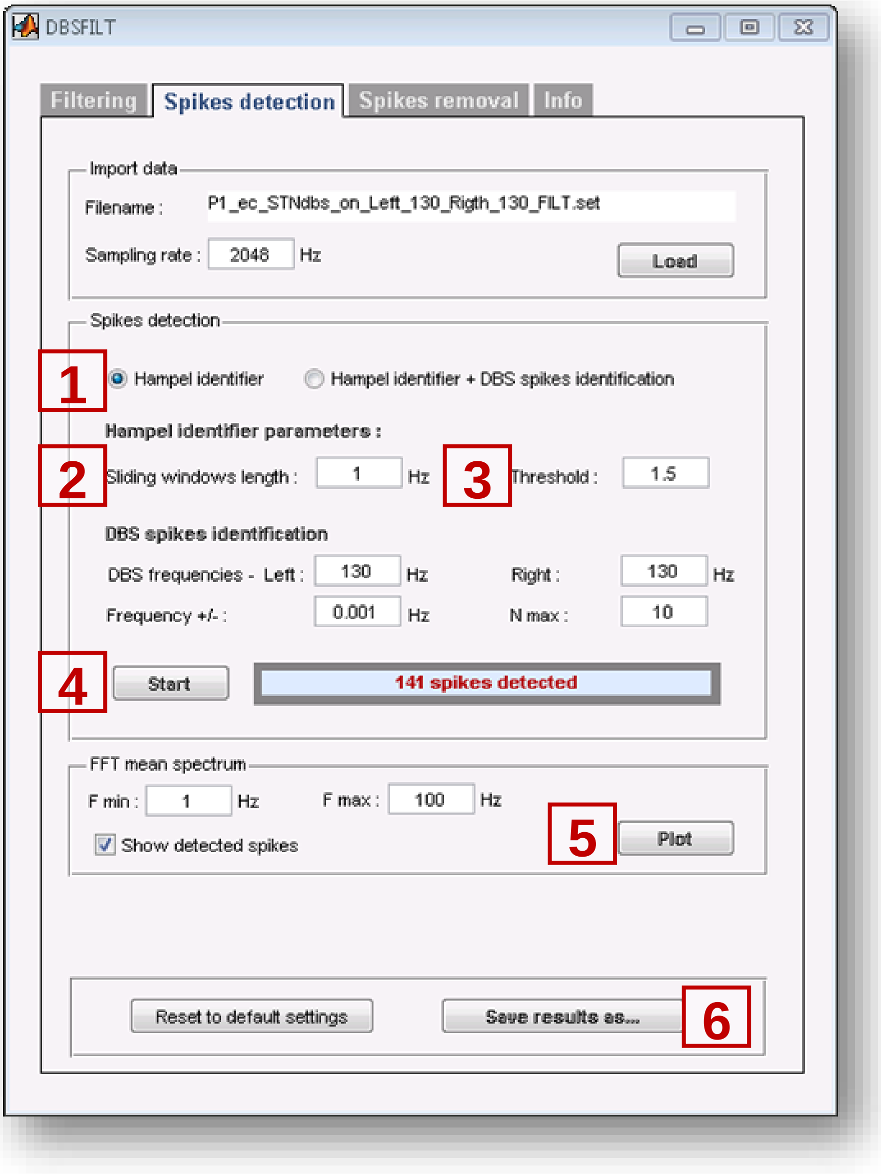
- [1]: Select "Hampel identifier"
- [2]: Choose the windows length (in Hz) where spikes will be detected as outliers.
- [3]: Choose the threshold for the Hampel identifier.

*Considering $f_{spike} \in [f_1; f_L]$ where L is the windows length. $f_{spike}$ is considered as an outlier if :*

$$\left|f_{spike} - median([f_1; f_L])\right| > T \times 1.4286 \times median([f; f_L])$$

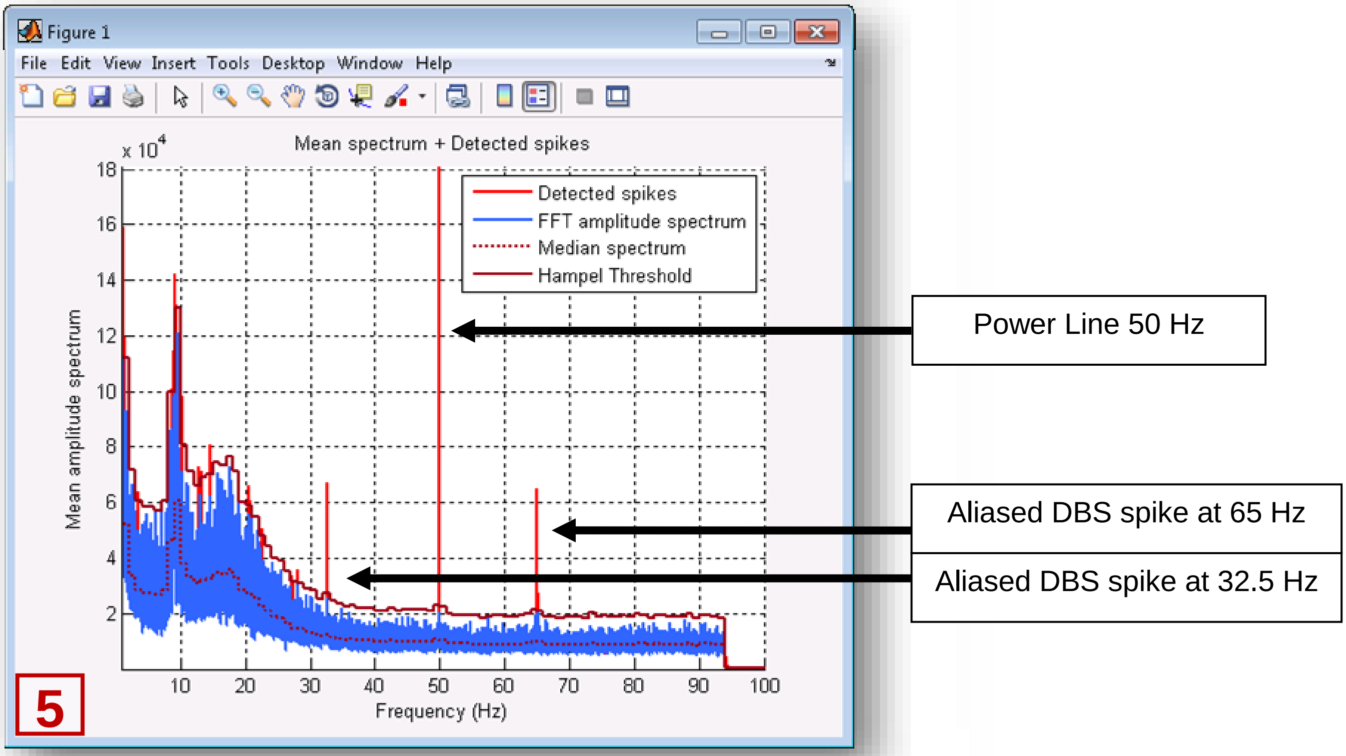
*Where T is the chosen threshold.*

- [4]: Start detection.
- [5]: Show detected frequencies.
- [6]: Save the results in a .spikes file.



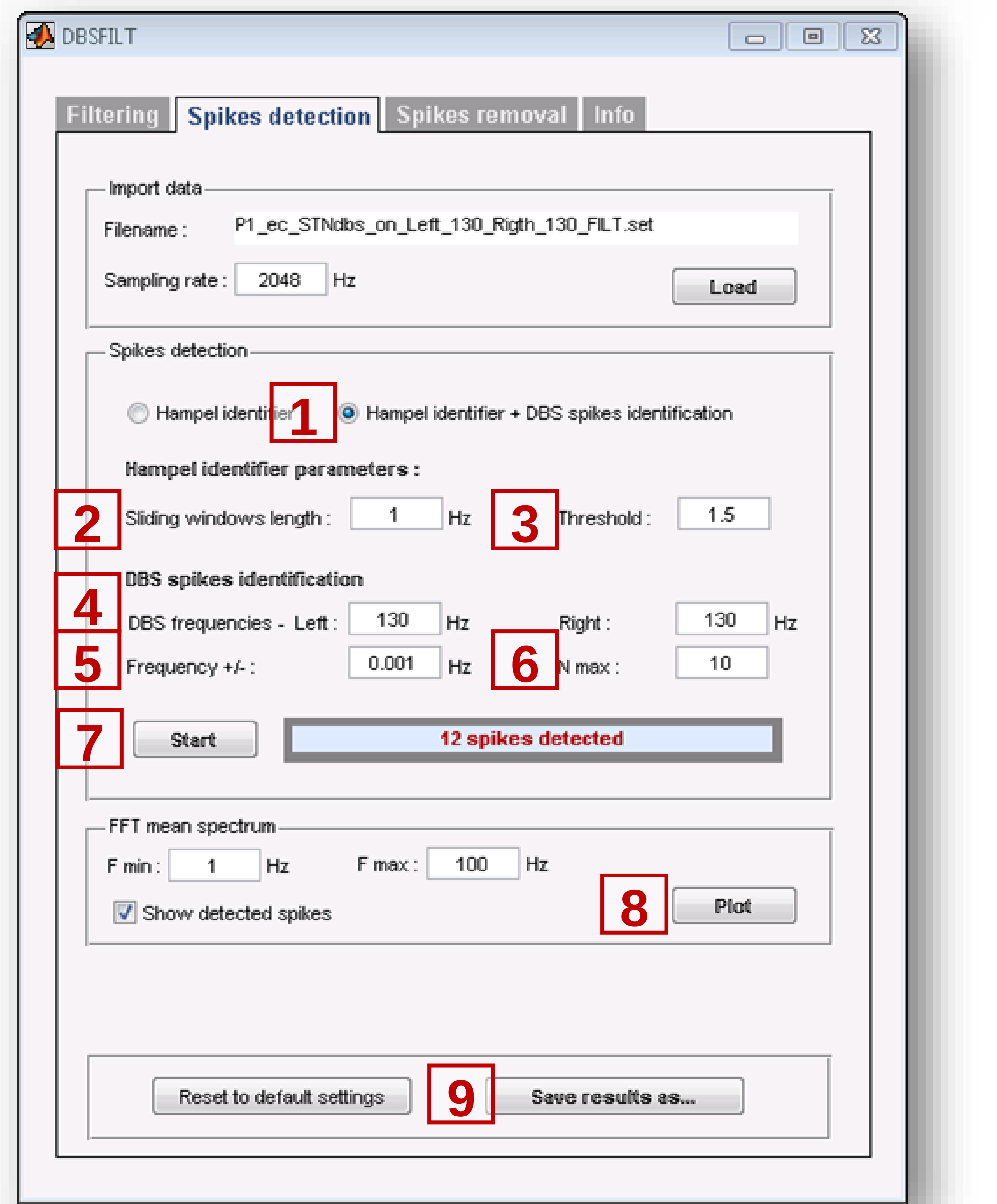
*Example of spikes detection with the standard Hampel filter :*

- Bilateral Pseudo-monopolar Stimulation of the STN at 130 Hz (spike width 90 µs)
- Windows length : 1 Hz
- Threshold : 1.5



***Step 2.2b*** – Spikes detection with the Hampel identifier using a supplementary routine to guide the filter towards the frequencies at which DBS artifacts are likely to occur.

- [1]: Select "Hampel identifier + DBS spikes identification"
- [2]: Choose the windows length (in Hz) where spikes will be detected as outliers.
- [3]: Choose the threshold for the Hampel identifier.
- [4]: Add left and right stimulation frequencies.
- [5]: Add a tolerance for the DBS frequency measurement (depending on the accuracy of the stimulation and the performance of the recording device).
- [6]: Choose the maximum number of DBS frequency harmonics to be considered.
- [7]: Start detection.
- [8]: Show detected frequencies.
- [9]: Save the results in a .spikes file.



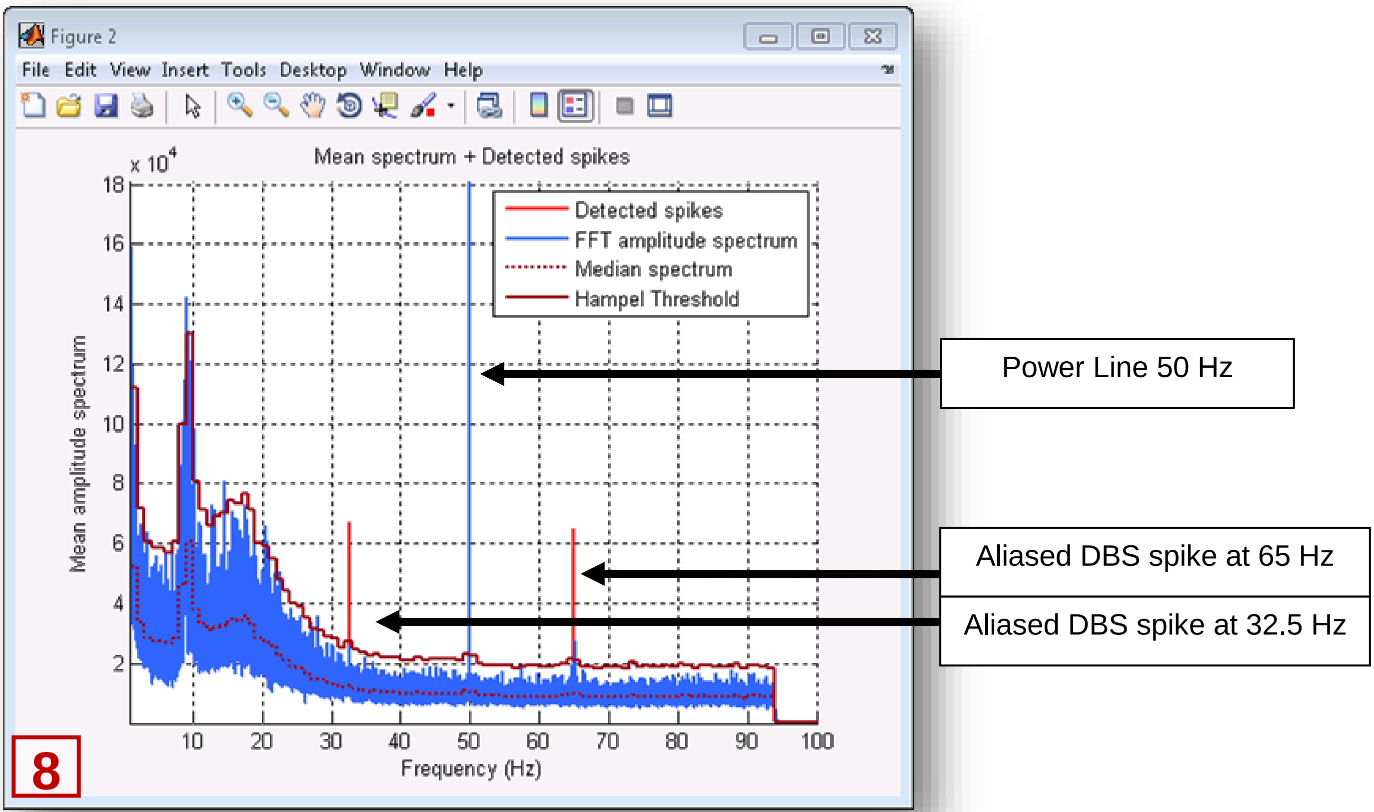
*__Example of spikes detection with the hampel filter and DBS spikes identification:__*

- Bilateral Pseudo-monopolar STN Stimulation at 130 Hz (spike width 90 µs)
- Windows length : 1 Hz
- Threshold : 1.5
- Frequency tolerance (Ftol) : 0.002 Hz
- Maximum number of DBS frequency harmonics considered (Nmax) : 10

For each spike detected with the Hampel identifier, the following equation is tested:

$$f_{spike} = h.\frac{f_{DBS}}{n}\ ; \quad [h \pm Ftol] \in \mathbb{N}^{+}; \quad [n < Nmax] \in \mathbb{N}^{+}$$

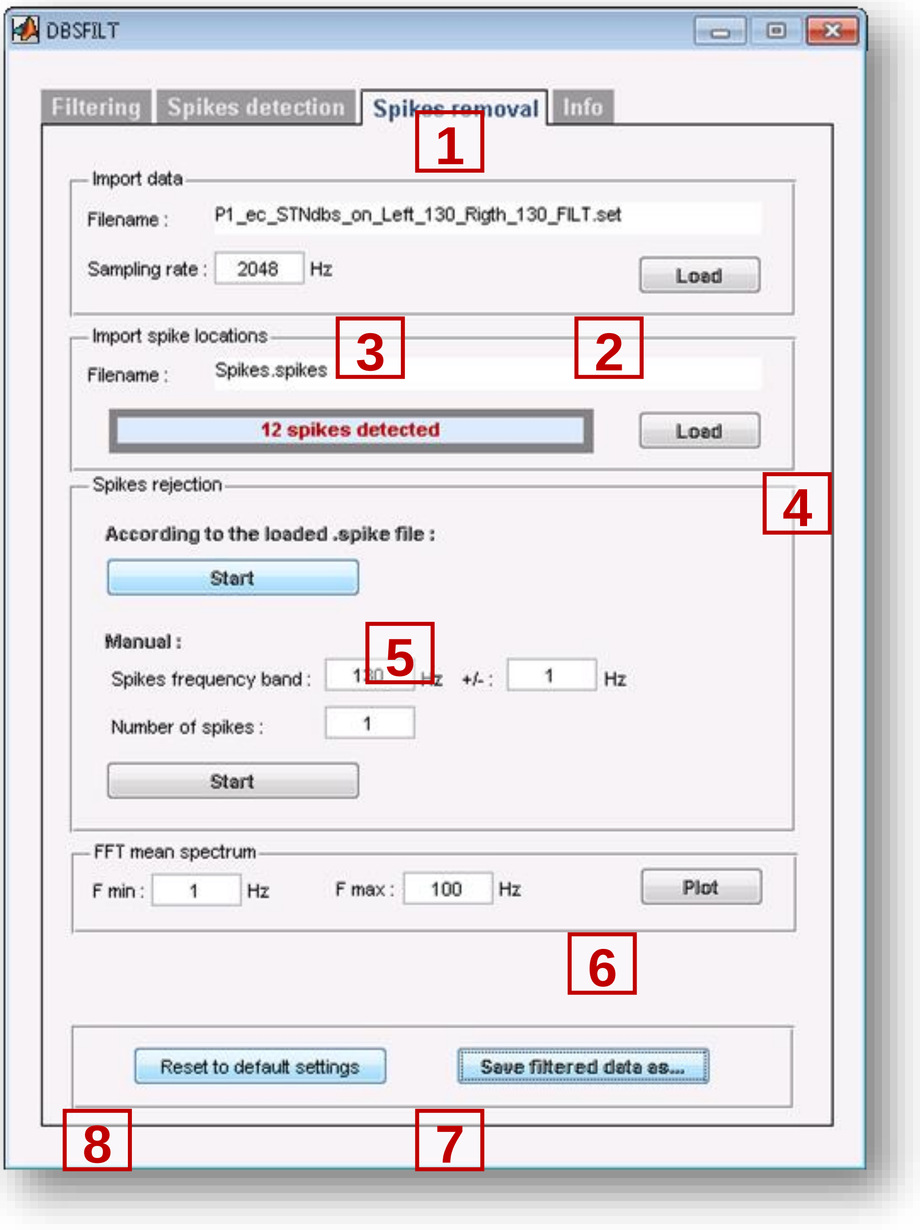
If the equation is true, the detected spike is identified as an aliased DBS frequency. As a consequence, the 50 Hz power line spike and powerful low frequency neuronal activities are no longer considered as artifacts.



## 6- DBS artifacts filtering:

### – Step 3, spikes removal.

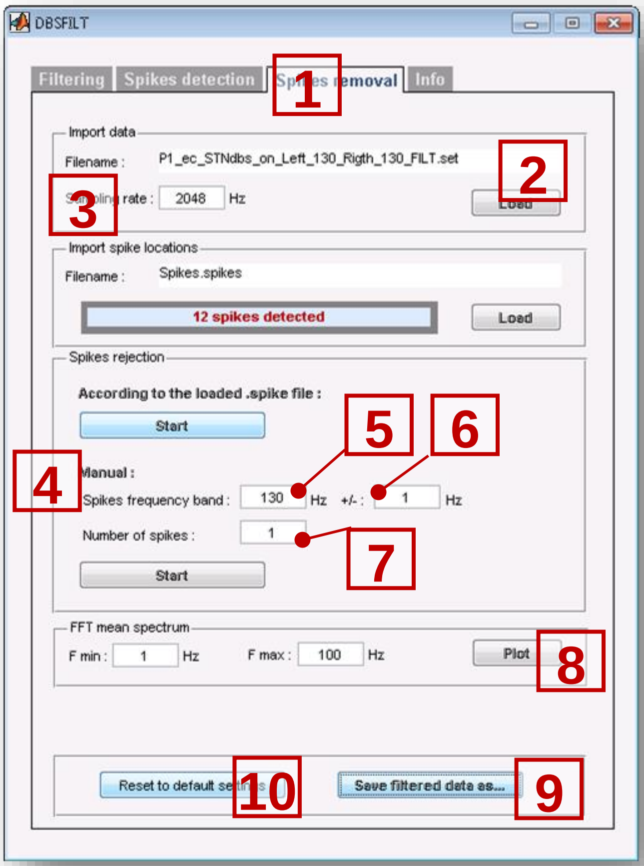
- Select the ‘Spikes removal’ tab [1], load the appropriate file [2] (generated in Step 1) and, if necessary, add the sampling rate [3].
- Load the .spikes file created during Step 2 (2.2, spikes detection) [4].
- Launch DBS spikes interpolation [5].
- Plot the FFT mean spectrum of the filtered data. [6]
- Save the filtered data [7] and clear all data in memory with the ‘reset’ button [8] if necessary.



## – *Optional: Manual DBS filtering by rejection of matched sinusoids.*

DBS artifacts can also be suppressed/attenuated by the rejection of matched sinusoid* at a manually selected frequency (Sun et al. 2014) :

- Select the 'Spikes removal' tab [1], load the appropriate file [2] and, if necessary, add the sampling rate [3].
- In the 'Spikes rejection - Manual' section [4] – Select the frequency of the matched sinusoid [5] and the size of the tolerance window [6]†.
- Select the number of *matched sinusoids* / *spikes* to be removed. [7] (= algorithm iterations).
- Plot the FFT mean spectrum of the filtered data. [8]
- Save the filtered data [9] and clear all data in memory with the 'reset' button [10] if necessary.




* : A *matched sinusoid* (temporal space) is named *spike* (frequency domain) in this software.
†: The algorithm finds and remove automatically the most powerful matched sinusoid in the specified frequency band.

## ***Technical note on the procedure of rejection of matched sinusoids:***

To remove a matched sinusoid, two methods have been implemented in the toolbox. The first method uses exactly the implementation described in the Sun et al. 2014 paper. It can be found in the toolbox at this location:

***fct\sig_proc\DBSremoval_ManualMatchedFilter\MatchedFilter.m***

This implementation generates a sinusoid *exactly* at the determined frequency, matches the sinusoid in amplitude and phase through a cross-correlation function in the time domain and subtracts the matched sinusoid. Since the recorded stimulation frequency differs from the theoretical stimulation frequency, an iteration parameter is used to ensure satisfactory removal of the artifact.

The second method is, from a conceptual point of view, strictly identical to the method of Sun et al. 2014. It suppresses a sinusoid matched in phase and amplitude with the noisy signal. But, the implementation is slightly different as it includes computational and practical optimizations. It can be found in the toolbox at this location:

***fct\sig_proc\DBSremoval_FFTguidedMatchedFilter\DBSFILT_ManualSpikesRemoval.m***

This implementation is directly accessible with the Graphical User Interface, but not the first one. There are many reasons for that:

1 – It takes the advantage of the computational efficacy of the fast Fourier transform (FFT) to achieve much better performance. By definition, discrete Fourier transform is the cross correlation of the input sequence of length N with a complex sinusoid at the frequency k/N. Therefore, it acts like a matched filter for that frequency with an optimized computational efficiency (e.g. – Smith 2007, 2018).

2- Since recorded and theoretical stimulation frequencies differ (Jech et al., 2006, Pamarti, 2010; Gaggl, 2013; Sun and Hinrichs 2016), a solution to optimize artifact detection is to automatically identify the frequency to be removed by adding a tolerance parameter to the algorithm. According to this procedure, the frequency of the matched sinusoid is automatically adjusted to the best suited frequency.

3- When the number of matched sinusoids that have to be removed is large (e.g. many iterations in the Sun et al. 2014 method, or several sinusoids removed with the FFT guided method), matched filters act as notch filters. Such drops in spectral-power at the frequencies of interest can cause substantial biases in future power-spectral analyses. The solution implemented here consists in replacing the removed sinusoid by another sinusoid with a variance interpolated from the power of the unaffected neighboring frequencies and with a random phase. This procedure reduces the strong bias in ON vs OFF power-spectral estimations that are likely to appear, especially when low-frequency aliased components are at stake (e.g., Beta band in Parkinson’s disease).