Wave clus

Automatic spike detection and sorting using wavelets and super-paramagnetic clustering.

Overview:

A large amount of research in neurophysiology is based on the analysis of extracellular potentials recorded with microwires that capture the action potentials (spikes) of neurons in their surroundings. For many applications it is crucial to know which spike correspond to which neuron, namely -spike sorting- and since recent acquisition systems allows the simultaneous recording of hundreds of channels, it is also important to do this automatically (or semiautomatically) and fast. Wave_clus is a fast and unsupervised algorithm for spike detection and sorting. Although it gives a first unsupervised solution, this can be further modified according to the experimenters' preference (semi-automatic sorting).

Distribution and reference:

Wave_clus is free (and therefore without any warranty) for any non-commercial applications. For any commercial application please contact the author (rqqg1@le.ac.uk). You can refer to this algorithm just by citing the paper where it is described:

Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering.
R. Quian Quiroga, Z. Nadasdy and Y. Ben-Shaul. Neural Computation 16, 1661-1687; 2004.

For non-technical references about spike sorting see:
Quick guide: spike sorting
Quian Quiroga, R.
Current Biology, Vol 22. R45–R46, 2012.

Spike Sorting
R. Quian Quiroga
Scholarpedia 2 (12): 3583. 2007

The original version of the software was developed by Rodrigo Quian Quiroga. Fernando Chaure, Matias Ison, Juan Martinez-Gomez, Carlos Pedreira and Hernan Rey contributed to the updated versions.

Requirements:

Wave_clus runs under Windows, Linux and Mac. It requires **Matlab 7.6 (R2008a)** or higher. It uses the function cluster.exe, provided by Eytan Domani, which is an executable that does the superparamagnetic clustering of the data. In the new release, the wavelet and the signal processing toolboxes are not necessary anymore.

Brief description of the code and parameters:

i) Parameters: The file set_parameters.m defines the values used by Wave_clus. In the user interface, the button "Set parameters" allows the user to edit the parameters or load their default values. The batch files (located in the folder "Batch_files") have an optional input variable to overwrite the settings in set_parameters.m for the current batch execution. In the following, parameters that can be set by the user appear in red.

ii) Spike detection: it first filters the continuous data between detect_fmin and detect_fmax. Then, it calculates an optimal amplitude threshold, which is set as stdmin times the estimated standard deviation of the noise (see the paper for details and advantages of this estimation). It also calculates a maximum threshold, stdmax, to avoid high amplitude artifacts. The parameter detection sets the threshold to be either positive, negative, or both. For each spike, w_pre datapoints before and w_post datapoints after the spike peak are stored. Spike alignment is done after interpolation of the spike shape with cubic splines. The interpolation is set with the parameters interpolation and int_factor. To avoid double detections, spikes should be separated at least a certain number of milliseconds, given by ref_ms. If the data doesn't include it, the sampling rate is set with the parameter sr. The parameter segments_length defines the length in minutes for segmenting the data in order to perform spike detection on each segment. This leads to performance improvements and avoids memory overloads. The code reads tmax seconds (or 'all' to read the whole file) of data starting from tmin.

iii) Feature extraction: it uses a selection of wavelet coefficients chosen with a Kolmogorov Smirnov test of Normality. The parameter inputs sets the total number of coefficients and scales is the number of scales for the wavelet decomposition. Alternatively, the code allows the use of the principal components as the spike features instead of wavelets with the parameter features (see the paper for comparisons of the different features). In some cases this will be the first step of the analysis, since many systems already give the detected spikes and not the continuous data.

iv) Clustering: uses super-paramagnetic clustering (SPC), an automatic clustering algorithm based on ideas from statistical mechanics developed by the group of Eytan Domani. See the paper for a description of its advantages for spike sorting, and papers by the group of Eytan Domani for other applications. The main variable to change with super-paramagnetic clustering is the 'temperature' (see paper). At low temperatures, all spikes will be assigned to a single cluster and at the high temperature limit; each spike will form a single cluster. The optimal temperature for clustering lies in between this two extremes (in the super-paramagnetic regime). In practice, the optimal temperature is set as the largest temperature for which a cluster with at least min_clus members appears. The code actually calculates results for all temperatures between mintemp and maxtemp in steps of tempstep. Changing the temperature is usually the main supervised correction done after clustering, but since results are already calculated for all temperatures in this range, it only implies loading and saving a different set of results.

v) Force membership: In many cases, points far from any cluster or in between two clusters are not assigned to any cluster. The function force_membership, (button Force in the GUI) will assign these points to the closest cluster unless they are too far (template matching). The algorithms in the Force_files folder were developed in collaboration with Casimir Wierzynski at Caltech. Clicking a second time the force button reverts results - also, in the new version the forced results are loaded from the batch files and clicking the force button shows the results without forcing. For each cluster, the number of spikes that were not assigned by template matching is shown in brackets.

vi) Template matching: Wave_clus also has the option of doing template matching after max_spk number of spikes, if the option match is set to 'y'. That means that Wave_clus will do SPC clustering for the first max_spk spikes (creating the associated templates for each cluster) and the rest of the spikes (if any) will be assigned via template matching. The principle of template matching is the same as the one used for Force_membership. Note that template matching can save a lot of time if there are too many spikes to be clustered. In particular, SPC starts taking too long for more than ~30000 spikes.

vii) Fix buttons: This option is for fixing a template before changing the temperature or pressing the force button. It is useful if a selection of clusters at different temperatures is needed. For example, you may want to keep, say, cluster 2 at temperature 0.5 and clusters 1 and 2 at temperature 1.0. Then you click on the fix button for cluster 2 at temperature 0.5 and then change the temperature to 1.0. The cluster you fixed will appear as a new cluster 3 at temperature 1.0. You may also don't want to use the force option for all clusters. Then, fix those clusters you want to keep and click the force button.

viii) Undo: This button restores the previous Wave_clus state.

ix) Merge: Merges two or more clusters. The clusters to be merged are selected with the fix buttons.Polytrodes: Wave_clus is also able to do spike sorting from polytrodes (including tetrodes), concatenating the spikes from different channels and taking them as a single shape for clustering purposes. First, write the name of the files you want to process as polytrode (in ascii format) in a text file named as polytrode1.txt. The spike detection is done by typing in the Matlab

command window the batch file Get_spikes_pol(1) which will produce the output file polytrode1_spikes.mat containing the spike times and shapes. Then execute Do_clustering(1) which produces the file times_polytrode1.mat. In case you may want to process for example 2 polytrodes you just have to type Get_spikes_pol(1:2) and Do_clustering(1:2).

You can visualize and change the clustering results with GUI by loading times_polytrode1.mat. Using the Plot_polytrodes button you will be able to visualize the clusters in each channel. The Plot all projections gives the peak amplitudes of the spikes for each channel against the others.

Data Input and Output:

The output of Wave_clus (obtained either using the Save clusters button in the GUI or the one given automatically by the batch files) is times_[filename].mat, which is a Matlab file containing the following variables: par (parameters used for clustering), spikes (a matrix with the spike shapes), inspk (a matrix with the features of the spike shapes) and cluster_class (a matrix with the clustering results). The variable cluster_class has 2 columns and *nspk* rows (*nspk* is the number of spikes). The first column is the cluster class, with integers denoting the clusters membership and a value of 0 for those spikes not assigned to any cluster. The second column is the spike times in ms.

Wave_clus can read Neuralynx, Blackrock, TDT, Plexon, Intan and ASCII files. If none of these corresponds to your data acquisition system, you should first convert the data to ASCII.

Getting started:

i) Copy the Wave_clus.zip file into your hard drive and unzip it.. If you want also to get the simulated data files (the ones used in the Neural Computation paper), download the file Simulator.zip in the Wave_clus/Sample_data subdirectory and unzip it there. This will create the subdir Simulator with all the files inside it. If you want to get the real human recording, download UCLA_data.zip in Wave_clus/Sample_data and unzip it there. This will create the subdir UCLA_data with a 30' recording inside it. Delete the .zip files once uncompressed. In Matlab go to the menu File/Set Path and add the directory Wave_clus with subfolders to the Matlab path. Copy the font in the subdir Wave_clus/Wave_clus_font into your windows/fonts/ directory. Now you are ready to use Wave_clus.

ii) For starting the GUI just type wave_clus in your Matlab command line. You can start playing with the simulated data used in the Neural Computation. You can load the simulated data (with the Load Data button) from the subdir Wave_clus/Sample_Data/Simulator and compare the results with the ones of the paper. You won't necessarily get exactly the same results, since the SPC clustering is a stochastic (i.e. not deterministic) method. Moreover, if you run the same

clustering twice you can get slightly different results (but in general is very robust). Check the file set_parameters in the dir Wave_clus/ for the parameters to use.

iii) The file CSC4 is an example of a 30' multiunit recording from a human epileptic patient. This data was collected at the lab of Itzhak Fried at UCLA, using a Neuralynx system (Tucson, Arizona). If you have a Neuralynx system, you can directly load your data. This is one nice feature of Wave_clus, that you can save all possible clustering options and then you just restore different options without recalculating everything (see how to use the batch files below). If you use the option Plot_average instead of Plot_all, results are restored much faster since plotting all the spikes may take too long. It may well happen that there is not a unique temperature for getting all the clusters (e.g. 1 cluster appears at temp=0.6 and another one at temp=0.7). If this is the case, you can easily obtain all the clusters by fixing cluster 1 at temp=0.6 and then go to the second temperature.

iv) Loading Matlab files with continuous data or spikes for clustering spike shapes that have already been detected (e.g. detected on-line by the acquisition system). The input data should be a Matlab file (extension .mat). It should be either a vector named data or a matrix named spikes (nr. of spikes x length(spike shapes)) plus a vector index with the spike times in ms. Before starting don't forget to set the proper sampling rate sr in the file set_parameters or inside the file. The sampling rate of the simulated data is 24KHz.

Batch files:

Since Wave_clus gives an unsupervised clustering of the data, it is possible to run batch processes that will go through several files and save the results. You can later restore the results in the GUI and change them, if needed, loading the times_[filename] file and the auxiliary SPC files.

The nice thing is that you don't need to calculate everything again. Wave_clus saves the results for all possible temperatures, so, changing the temperature (which is the main parameter to be changed, if needed) means just loading another set of results. There are 2 batch files: Get_spikes (which does the spike detection) and Do_clustering (which does the spike sorting). Get_spikes reads the continuous data and outputs a file [filename]_spikes with the variables index (the spike times in ms) and spikes with the spike_shapes. If you already have the spike shapes, you don't have to run Get_spikes, but you should have the same structure as the one saved by Get_spikes. Do_clustering inputs the [filename]_spikes files and does the clustering automatically. In this case, for getting the best out of Wave_clus spikes should be aligned having the maximum (minimum) of all them in the same sample. Results are saved in the file times_[filename].mat, which has the same structure as the one saved with the GUI. Although it will save and print results with the automatically chosen temperature, it stores results for all temperatures, so that it can be easily changed later with the GUI. Both Do_clustering and Get_spikes go through all the files specified in files.txt.

Comments and updates:

We really hope this algorithm will be useful for you. For any comments or suggestions please use: https://github.com/csn-le/Wave_clus/

We can't promise to reply, fix bugs or introduce suggestions quickly, but we'll try our best.

New features in version 2.5:

- Runs under Windows, Linux or Mac. Support both 32 and 64 bits architectures.
- Manual selection of clusters. Selecting a rectangle in any plot will create a new cluster with all the spikes with local extrema inside the selected rectangle.
- If the spikes have been previously detected (there is a XXX_spikes.mat file) or even sorted (there is a times_XXX.mat and data_XXX.dg_01 files), the GUI loads these results instead of recomputing everything. Delete such files if you want to recalculate the solution. Note that when doing this, if the parameters used to create that solution are stored in the spikes/times files, they will overwrite the values appearing in set_parameters.m.
- Can open times_XXX.mat even without the SPC files (although moving through the temperature map will not be possible).
- Automatic load from different types or raw data. The "Raw_data_reader" folder has the functions for importing raw data. The first letters of the .m files are the extensions that the file reads (Wave_clus use that to dynamically know what it can read).
- Only one batch file and set_parameters for all data types.
- Parallel processing of batch files.
- Save and Load Gui state.
- Many bug fixes and performance improvements.
- The "tools" folder has the codes we use for parsing the raw data files into individual channels. Wave_clus needs only one channel and some formats store all the data in a single file. We create new file extensions for the single channel files.
- Supported file formats: mat (Matlab), ncs (Neuralynx), nse (Neuralynx), int (Intan), NSx (Blackrock), tdt (TDT) and PL2 (Plexon)

FAQs:

- What should I change if I don't like the automatic results?
The main parameter to change is the temperature. You can also use the force button to add some of the non-clustered spikes to the already defined clusters.

- How can I run Wave_clus faster?
There are few tricks to run faster. First, if you have too many spikes, you can do clustering on a first segment that will define the clusters, and assign the remaining spikes via template matching (the templates being the clusters defined in the first segment). This is set with the parameter max_spk. You can also use the option plot_average instead of plot_all, since plotting all spike shapes may take too long. If you just want to check a short segment of the file you can do this using the parameter tmax. If the functions of FilterM are in the path, Wave_clus will use them for reducing the time required for filtering the raw signal.

- How can I avoid high temperatures?
Sometimes Wave_clus chooses high temperatures that tend to overcluster the data. To avoid picking up high temperatures, you can set the parameter min_clus to a larger number.

- What are the Accept / Reject?
Accept saves the clusters by assigning a corresponding integer (say 1 for the first cluster, etc.). Reject gives the cluster a value of 0; i.e. it will merge it with the non-clustered spikes.

- What are the numbers appearing at the top and bottom of each cluster?
The top number is the total number of spikes for the cluster, with the ones not assigned through template matching appearing in brackets. The bottom one is the number of spikes with an inter-spike-interval (ISI) of less than 3ms. A relatively large number of spikes within 3ms ISI is a sign of a multiunit cluster.

- What happens if I have more than 3 clusters?
In this case as many extra figures as necessary will be generated.

- Why the spike features do not show the clusters well separated?
Wave_clus is plotting only the first 2 wavelet coefficients. It may well happen that other coefficients will separate the clusters even better. You can actually plot 45 projections, corresponding to 10 wavelet coefficients, with the Plot all projections button.

- Can I edit the plots?
Yes, you can edit, zoom, copy and paste, etc each plot using the Tools menu at the top of the GUI.

- What is the set_parameters button?

You can choose the parameters or load the default values. You should do that before loading the data.

- I get a memory overflow error when loading the data, what can I do?
Just decrease the segments_length for loading the data in smaller pieces.