Journal of Neuroscience Methods 155 (2006) 207–216

JOURNAL OF NEUROSCIENCE METHODS

www.elsevier.com/locate/jneumeth

Klusters, NeuroScope, NDManager: A free software suite for neurophysiological data processing and visualization

Lynn Hazan[1], Michaël Zugaro[1], György Buzsáki*

Center for Molecular and Behavioral Neuroscience, Rutgers, The State University of New Jersey, Newark, NJ 07102, USA

Received 12 August 2005; received in revised form 6 January 2006; accepted 10 January 2006

Abstract

Recent technological advances now allow for simultaneous recording of large populations of anatomically distributed neurons in behaving animals. The free software package described here was designed to help neurophysiologists process and view recorded data in an efficient and user-friendly manner. This package consists of several well-integrated applications, including NeuroScope (http://neuroscope.sourceforge.net), an advanced viewer for electrophysiological and behavioral data with limited editing capabilities, Klusters (http://klusters.sourceforge.net), a graphical cluster cutting application for manual and semi-automatic spike sorting, NDManager (http://ndmanager.sourceforge.net), an experimental parameter and data processing manager. All of these programs are distributed under the GNU General Public License (GPL, see http://www.gnu.org/licenses/gpl.html), which gives its users legal permission to copy, distribute and/or modify the software. Also included are extensive user manuals and sample data, as well as source code and documentation.
© 2006 Elsevier B.V. All rights reserved.

Keywords: Spike sorting; Cluster cutting; Data mining; Unit activity; Local field potentials

1. Introduction

In recent years, remarkable technological advances have allowed neurophysiologists to record from large ensembles of anatomically distributed neurons in behaving rodents and primates (Buzsáki et al., 1992; Wilson and McNaughton, 1993; Wilson and McNaughton, 1994; Hampson et al., 1999; Hoffman and McNaughton, 2002; Csicsvari et al., 2003; Nicolelis et al., 2003; Buzsáki, 2004; Nicolelis, 1998; Eichenbaum and Davis, 2001). However, visualizing and processing the large amounts of data generated by modern recording systems requires efficient computer software. The free software package presented here consists of several well-integrated applications and tools designed to assist the experimenter in extracting and exploring brain signals, starting from raw (wide-band) or preprocessed (action potentials and local field potentials) signals typically recorded by hardware acquisition systems. The integrated package has been successfully used in recent studies (Khazipov et al., 2004; Zugaro et al., 2005).

The programs presented below are designed to process and explore data collected in experiments ranging from acute recordings in anesthetized animals to complex chronic recordings where brain signals are recorded from freely moving animals as they perform behavioral tasks in automated apparatuses. Thus, in the most complex cases, the data can consist of electrophysiological signals (action potentials and local field potentials), behavioral events (e.g., crossing of photodetectors, reward delivery) and video recordings (e.g., position tracking).

Of the three types of data, processing of electrophysiological signals is the most challenging. Extracellular electrodes typically record action potentials emitted by several nearby neurons, many of which are relegated as indiscrimnable ‘noise’, and occasionally artefacts generated by muscle activity, surrounding electrical devices and other sources.

Processing of electrophysiological data requires three steps (Fig. 1). First, putative action potentials (‘spikes’) must be detected and extracted from the wide-band signals in hardware or software. This is usually done first by high-pass filtering and thresholding, then by extracting an adequate number of voltage samples (e.g., corresponding to a time window of a spike) each time a predetermined threshold is crossed. Thus, each spike is described by a vector, the components

* Corresponding author. Tel.: +1 973 353 1080; fax: +1 973 353 1820.
E-mail address: buzsaki@axon.rutgers.edu (G. Buzsáki).
[1] These authors have contributed equally to the present work.

0165-0270/$ – see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.jneumeth.2006.01.017

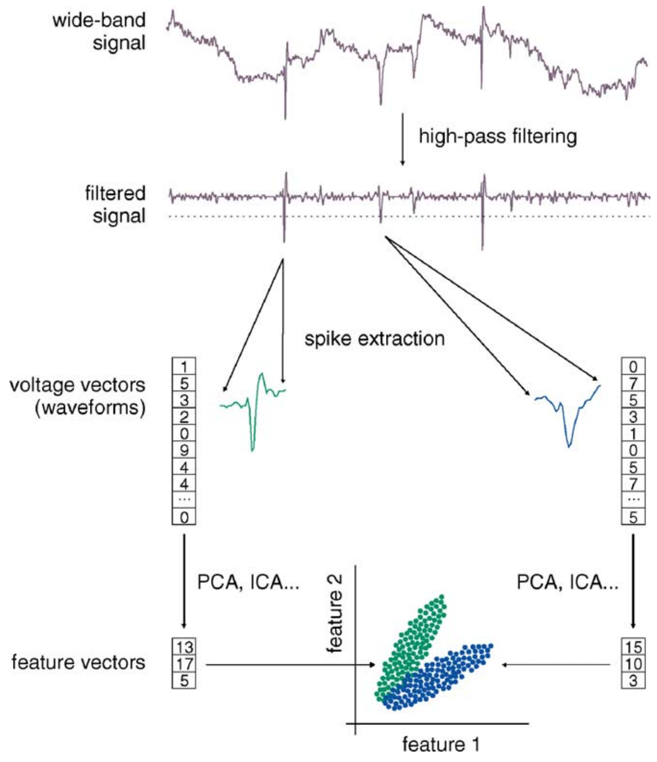


Fig. 1. Processing of electrophysiological spike data. First, wide-band continuous brain signals are high-pass filtered and thresholded to detect putative action potentials ('spikes'); waveforms are extracted around the peak of each spike. Second, the high-dimensionality of the waveforms is reduced for subsequent sorting, using e.g., principal component analysis. Finally, the resulting feature vectors are sorted. The goal of this process is to group the action potentials emitted by each single neuron into distinct clusters. Ideally, each recorded action potential is assigned to the neuron that emitted it.

of which are successive voltages in time. The second step is feature extraction. Although complete voltage vectors ('waveforms') are the most accurate description of the spikes, they are rarely appropriate for subsequent processing because of their high dimensionality. There are several methods to drastically reduce the number of components per spike while retaining most of the relevant information. These include principal component analysis (PCA, Abeles and Goldstein, 1977), independent component analysis (ICA, Jutten and Herault, 1991), factor analysis, and other related methods. The third step is 'spike sorting', where spikes are tentatively assigned to the individual neurons that have emitted them. This results in grouping spikes in different 'clusters' corresponding to different putative neurons (hence, this process is also referred to as 'cluster cutting'). Spike sorting is most efficiently performed by combining semi-automatic and manual approaches (Harris et al., 2000).

Action potentials of single cells are embedded in networks and related to behavior and the ultimate goal of spike detection is to reveal these relationships. Therefore, spikes should be displayed and compared to spikes of other neurons, local field potentials and behavioral events. To facilitate these analyses, we describe a number of computer applications and tools, which allow for processing and visual exploration of the data before subsequent quantitative analyses.

2. The software suite

What makes the package described here different from other existing offerings is threefold. First and foremost, it is free software distributed under the GNU General Public License (GPL, see http://www.gnu.org/licenses/gpl.html). With this license, our users are explicitly granted legal permission to copy and redistribute the software, as well as to modify it (or have someone modify it) using the source code. We therefore distribute the software both in binary and source forms (care was taken to develop high-quality, clear and documented code). We hope this will help form a community of interested investigators willing to contribute to the project. The second difference is that our software can be downloaded from the internet at no cost. The third difference is that because our software was developed within a neuroscience laboratory rather than by an external company, the feature set was directly chosen and defined by experimenters. This resulted in a package with numerous and relevant advanced features (such as the Error Matrix View or the Trace View in Klusters). In addition, constant and direct user feedback ensured that the programs featured easy and efficient user interfaces: intuitive and flexible layout of the graphical elements, numerous configurable keyboard shortcuts and consistent keyboard navigation, optimized display speed, highly responsive interface via multithreading, etc.

Our package runs on GNU/Linux and MacOS X and is expected to run on any Unix-like system that includes the KDE libraries and libxml2. We provide installable binary packages for Debian-based distributions (Debian, KNOPPIX, Kubuntu, etc.) and SUSE. Alternatively, the applications can be built from source (they are known to run on e.g., RedHat and MacOS X with Fink). Detailed information is available at the respective websites.

3. Data formats and preprocessing

Contrary to many data acquisition and processing programs, our software does not use a single file with a complex structure, but a collection of very simple files. This ensures that files are easy to read from and write to, and can thus be manipulated using any data analysis package without requiring complex import and export filters. There are dedicated files for continuous brain signals (wide-band .dat, local field potentials .eeg, high-pass filtered .fil, etc.), spike waveforms (.spk), feature vectors (.fet), spike clusters (.clu), behavioral events (.evt) and position tracking (data file formats are described in Supplementary Figs. 2 and 3). To reduce disk usage, potentially large files (continuous brain signals and spike waveforms) contain multiplexed binary data. All other files contain ASCII format data, making them easy to manipulate by standard Unix file utilities. Files are homogeneous and do not contain headers. All the relevant information (number of channels, sampling frequency, spike waveform length, date and comments, etc.) is stored in a common parameter file in XML format. This standard, well-supported, self-described format allows for easier extensibility for future versions of the software.

In many cases, data needs preprocessing before analyses can be performed. An example is spike extraction from continuously recorded wide-band signals. Continuous recording of wide-band signals allows for a complete post-hoc replay and exploration of the original, unprocessed brain signals recorded during the experiment. Other examples of data preprocessing include extracting from a video stream the position of the head lights carried by the animal, concatenating multiple recording files, or simply converting data files from a proprietary format to one of our open formats.

4. Klusters: a graphical spike sorting application

Klusters is a graphical application for manual spike sorting. It can be used either to improve the output of automatic clustering or to manually cluster raw data. The initial set of features was inspired by the cluster cutting program sgclust by J. Csicsvari (unpublished).

Klusters works with a spike waveform file (.spk) and a feature file (.fet), and optionnally a cluster file (.clu) produced by an automatic clustering program (Harris et al., 2000) such as KlustaKwik (K.D. Harris, http://klustakwik.sourceforge.net). For data recorded continuously, the wide-band recording file (.dat) can also be used to display raw traces. Once the clusters have been manually created or refined by the experimenter, upon saving a cluster file (.clu) is created.

After loading the files, Klusters displays an overview of the data. This includes several graphical elements referred to as ‘views’, namely a cluster view, a waveform view and a correlation view (described in detail below). It is possible to work with several displays in parallel, each of which can flexibly combine different numbers and types of views arranged in custom layouts and showing different subsets of the data (Fig. 2). On the left side of the main window is the palette where each cluster is represented as a colored square. Clusters selected in the palette are shown simultaneously in all the views of the currently active display. Notice that in order to ensure rapid and easy identification, individual clusters are drawn using the same customizable colors throughout the application (palette and views).

4.1. Cluster views

For viewing and editing purposes, Klusters provides cluster views displaying two-dimensional projections of the feature vectors. To help visualize the data in multiple dimensions, any number of cluster views can be combined to display different projections simultaneously (Fig. 3). Several editing tools are available. These allow for direct manipulation of whole spike clusters as well as arbitrary subsets of points enclosed within user-defined polygons.

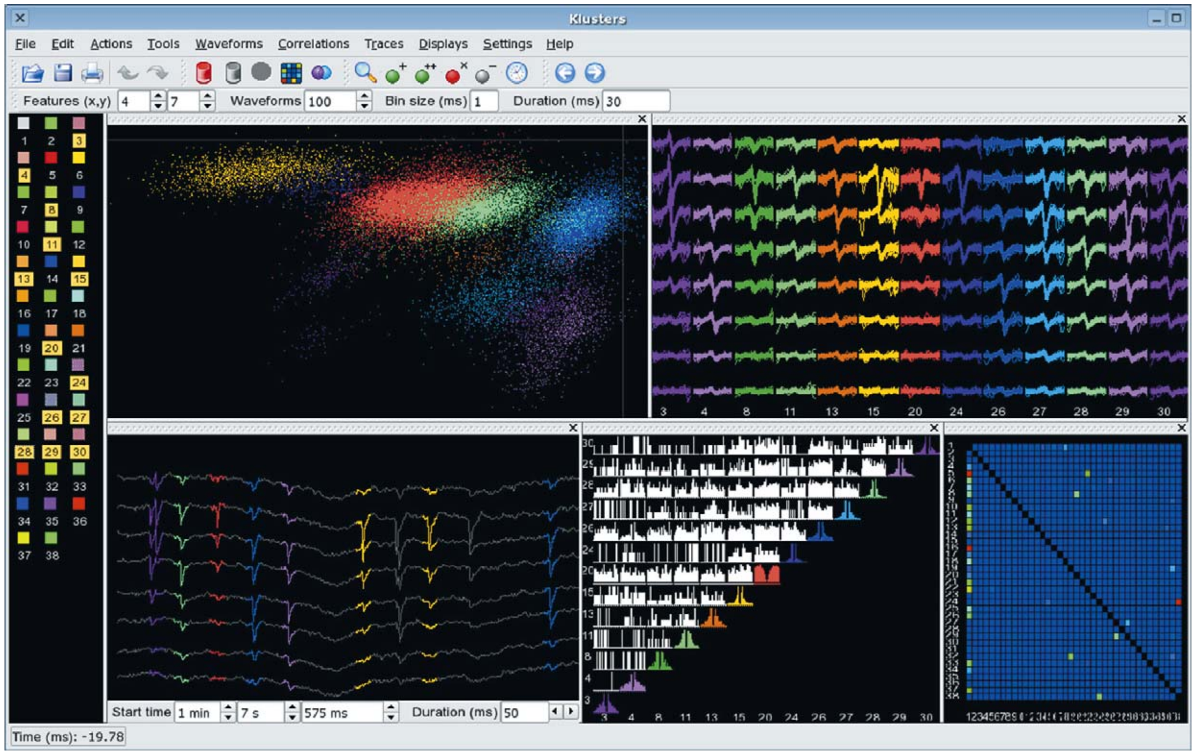
Klusters provides several specific tools to discard ‘artefacts’ and biological signals designated as ‘noise’ (e. g., background small amplitude multiunit activity). Discarded spikes are never actually deleted from the data files; rather, by convention, artefacts and noise are assigned to clusters 0 and 1, respectively. Klusters provides additional tools to create new clusters or correct existing clusters. Although automatic spike sorting should ideally yield one distinct cluster for each neuron, in practice, two different kinds of systematic error can occur, which require manual correction: ‘overclustering’, where spikes emitted by a single neuron are split in multiple clusters, and ‘underclustering’, where individual clusters contain spikes from multiple neurons. To correct for overclustering, multiple clusters can be grouped together. A special case occurs when one or more electrodes drift during the course of the recording session. In this case automatic spike sorting typically splits the spikes in several clusters because spike amplitudes change as a function of time. Selecting time as one of the projection dimensions and carefully inspecting the resulting clusters helps correct for such errors (Fig. 4). To correct for underclustering, one can split existing clusters by manually selecting arbitrary sets of points. These are extracted from their current clusters and, depending on the tool, either grouped together in a single new cluster, or assigned to one new cluster for each original cluster.

4.2. Waveform views

Assessing whether a cluster is contaminated by action potentials of other neurons or artefacts, or determining whether two or more clusters actually correspond to a single neuron, is facilitated by inspection and comparison of spike waveforms (Supplementary Fig. 1). Klusters provides waveform views where the waveforms for the currently selected clusters are displayed as colored traces (side by side, or overlaid). Because clusters can contain hundreds of spikes, by default only a subset of the waveforms are actually displayed (in order to reduce display time and memory usage). There are two possible selection modes: the view displays either a user-defined number of waveforms evenly spaced in time, or all waveforms occurring within a customizable time frame. Waveform means and standard deviations can also be displayed.

4.3. Correlation views

Autocorrelograms and cross-correlograms plotted in correlation views (Fig. 5) provide invaluable information for assessing the success of spike clustering. For instance, well isolated clusters contain spikes emitted by a single neuron, and thus their autocorrelograms show a clear refractory period (McCormick et al., 1985; Fee et al., 1996; Csicsvari et al., 1998; Harris et al., 2000). Conversely, autocorrelograms which do not show a clear refractory period correspond to ‘noisy’ clusters, i.e. clusters which combine spikes emitted by multiple units. Auto- and cross-correlograms also help finding instances of single neurons erroneously split across multiple clusters (cross-correlograms resembling the respective autocorrelograms) and identifying successive spikes within complex spikes (one-sided cross-correlograms): for example, the asymmetry and common refractoriness of the crosscorrelograms between clusters 11 and 15 are strong indications that these subclusters should be combined since they represent subsequent spikes of complex spike bursts. In addition, the correlation view is also used for screening monosynaptic excitatory and/or inhibitory connections between cell pairs, characterized by short-latency, large



Features (x,y) 4 7 Waveforms 100 Bin size (ms) 1 Duration (ms) 30

Start time 1 min 7 s 575 ms Duration (ms) 50

Time (ms): -19.78

Fig. 2. Klusters. An overview, showing a cluster view (two-dimensional projection of the feature vectors, top left view), a waveform view (top right view), a trace view (spikes highlighted on wide-band brain signals, bottom left view), a correlation view (auto- and crosscorrelograms, bottom middle view), and an error matrix view (color-coded matrix of mean identification error probability, bottom right view). Individual views can be interactively moved around in the display, and more views can be added (e.g., additional cluster views using different projection features, and additional correlation views displaying different time scales). For rapid identification, data from single clusters are represented using the same color throughout the interface, including the selection palette (left panel). Several tools are shown in the tool bar (icons): delete artefact cluster(s) (red trash can), delete noise cluster(s) (grey trash can), update error matrix (colored matrix), group clusters (intersecting purple and blue spheres), zoom (magnifier lens), new cluster (green sphere with a plus sign), split clusters (green sphere with two plus signs), delete artefact spikes (red sphere with a cross sign), delete noisy spikes (grey sphere with a minus sign), select time (clock), previous and next spikes (blue arrows). Several view parameters can be adjusted in the various text fields: projection features, number of waveforms, correlogram bin size and half duration, trace start time and duration (data provided by David Robbe).

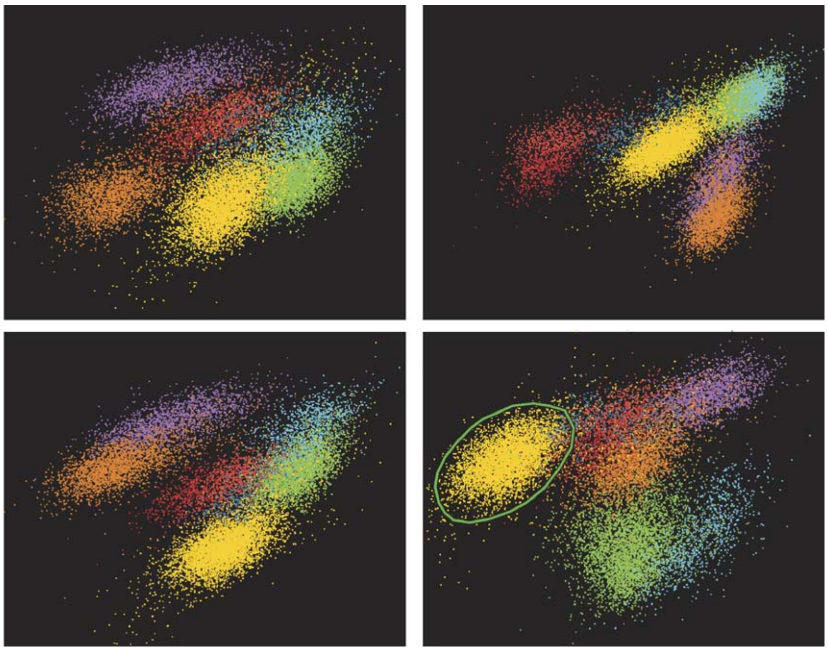


Fig. 3. Cluster views. Each view shows different two-dimensional projections of the feature vectors. Different clusters are represented by different colors. Any number of cluster views can be used simultaneously in order to display different projections of the data side-by-side. Editing tools allow for the creation of new clusters from points enclosed in user-defined polygons (green polygon in lower right panel), as well as grouping or deletion of existing clusters.

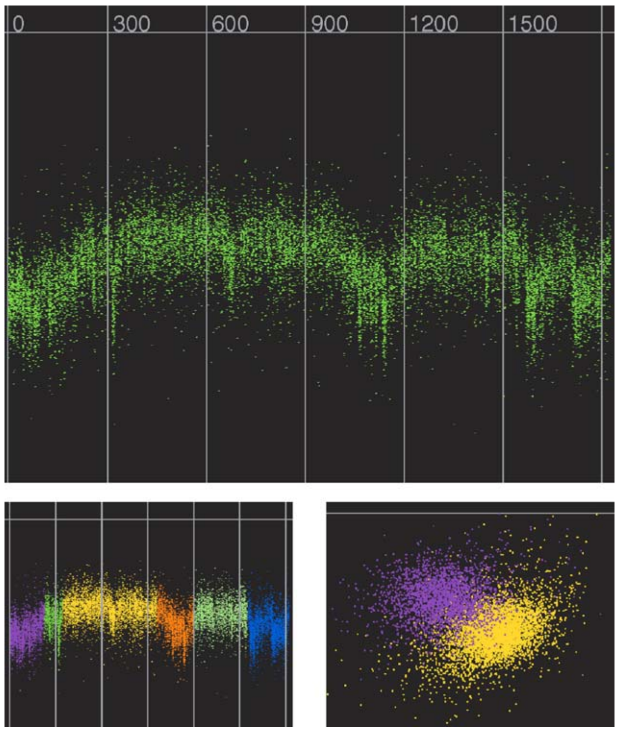


Fig. 4. Time axis in cluster views. Using time as one of the projection dimensions shows how spike features change over the course of a recording session. Top: this cluster shows evidence for electrode drift, as indicated by the magnitude shifts of the feature on the *y*-axis. Such drifts may occur when the electrode is located very close to the neuron and even relatively small relative movements can induce noticeable spike waveform changes (e.g., in amplitude). When this happens, the automatic clustering algorithms typically split the spikes across multiple clusters (bottom left), which can misleadingly appear distinct in two-dimensional projections (bottom right). Here, using time as one of the projection dimensions helps identify such misidentification.

amplitude, 1–3 ms wide bins or spike suppression (Csicsvari et al., 1998; Barthó et al., 2004).

4.4. *Error matrix view*

Although visual inspection of auto- and crosscorrelogram is necessary for the adjustment of clusters, quantification of such errors is advantageous when large numbers of clusters are formed. To this end, Klusters provides an error matrix view, a graphical representation of a statistical measure of cluster similarities (Fig. 2). This view indicates for each pair of clusters the mean probability that the spikes in the first cluster actually belong to the second cluster, using the same estimation method as the Classification Expectation Maximization (CEM) algorithm (Celeux and Govaert, 1992) implemented in KlustaKwik.

4.5. *Trace views*

Provided that continuous wide-band signals were recorded during the experiment, Klusters can use the corresponding data file (.dat) to display trace views where brain signals (spikes and local field potentials) are shown in time (Fig. 2). Exploring raw data allows for direct assessment of statistical properties evidenced e.g., in auto- and cross-correlograms, and helps detect spurious effects or confirm likely hypotheses (for instance regarding complex spike bursts). Trace views feature a reduced set of the functionalities available in NeuroScope such as spike browsing (see below).

4.6. *Automatic reclustering*

Although Klusters features a number of tools to correct for underclustering, in most cases manually splitting clus-

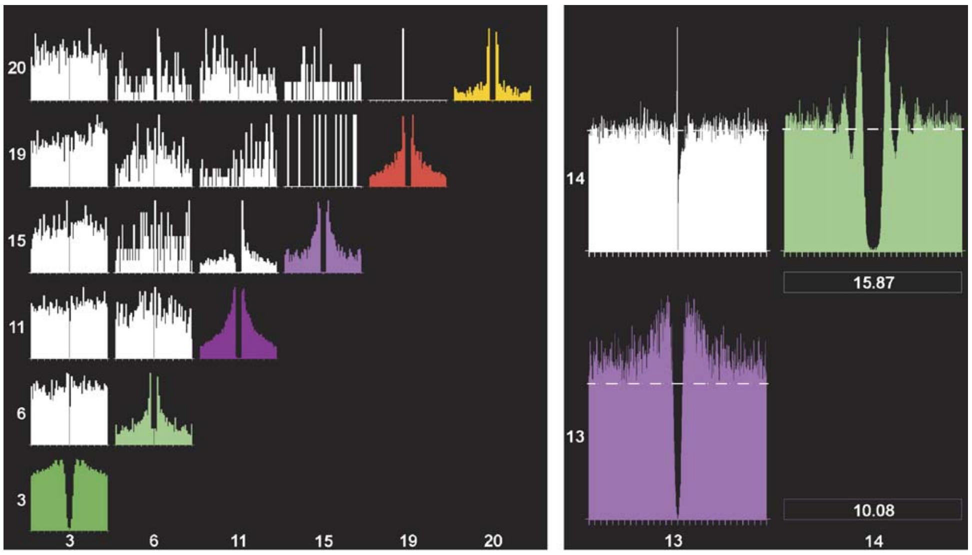


Fig. 5. Correlogram views. Autocorrelograms (colored histograms) and cross-correlograms (white histograms) help determine isolation quality. For instance, cluster 6 has a noisy refractory period, indicating poor isolation. Similar refractory periods in the auto- and the cross-correlograms together with a single-sided peak in the cross-correlograms for clusters 11 and 15 reveal splitting of spikes of bursty neurons. Autocorrelogram 3 with its broad shoulder is indicative of a putative fast spiking interneuron. The asymetric cross-correlogram for clusters 13 and 14 (right) suggests a bidirectional monosynaptic connection between excitatory and inhibitory neurons. Displaying a longer time scale (right vs. left) can reveal firing rythmicity (e.g., for cluster 14). Notice the average firing rate optionally indicated in the color boxes and as dashed horizontal lines.

ters involves arbitrary and potentially biasing choices (a typical example is cluster 'shaving', where peripheral points are removed from a cluster). To allow for unbiased reclustering of a set of existing clusters, Klusters can interactively run an automatic clustering program on a subset of the data. This is usually a computing intensive task, which can require long processing time, and should thus be performed on a reduced number of clusters. In particular, this feature should not be used as a graphical front-end to initial automatic cluster cutting over the whole data. For large files, this can run for up to several days on current computer hardware.

4.7. Exporting as vector graphics

Once optimal clusters have been determined, it is often useful to export waveform traces, spike cluster plots and cross-correlograms for presentations or manuscripts. Although this can be achieved by simply using a screen capture utility, doing so results in bitmapped images that do not scale or print well. Instead, Klusters can export high-quality graphics to PostScript (PS) or Portable Document Format (PDF) files. These standard vector graphics formats can then be imported in any drawing application for further editing.

5. Neuroscope: an electrophysiological and behavioral viewer

NeuroScope is a viewer for continuously recorded signals (e.g., wide-band or local field potentials), spiking activity, and behavioral events.

NeuroScope is not a data analysis program; rather, it allows for the easy and efficient inspection of raw data. Although statistical analyses and plots provide synthetic views of data properties, direct examination of raw data often provides invaluable insight into the fine structure underlying these properties, helps avoid spurious effects, and may generate hypotheses for subsequent quantitative analyses. NeuroScope was specifically designed to efficiently handle large amounts of data (dozens of channels recorded at high sampling rates, plus dozens of neuronal spike trains, and video tracking).

NeuroScope works with a continuous recording file (.dat, .eeg, .fil, etc.). Optionally, it can also display unit activity loaded from spike timings files (.res) and cluster files (.clu), as well as position tracking data from a position file (.whl), and behavioral events from one or more event files (.evt). Although NeuroScope is mainly a viewer application, it features limited editing capabilities allowing for modification of event files (.evt), such as adding markers to visually identified events.

NeuroScope displays the data in a trace view (electrophysiological signals) and optionally a position view (position tracking) combined in a display (Fig. 6). Similarly to Klusters, it is possible to work with several displays in parallel, each of which can show different subsets of the data. On the left side of the main window is the palette where channels, units and events are represented as colored icons in dedicated tabbed layouts. Again, rapid and easy identification is ensured by drawing individual elements using the same customizable colors throughout the application (palette and views).

5.1. Browsing continuous brain signals

Any number of continuously recorded channels can be displayed in custom colors and arrangements (Fig. 7). Data browsing combines direct access to specific points in time, and step-by-step replay of recordings across time. The duration of the data displayed in the view can be adjusted (doubled, halved,

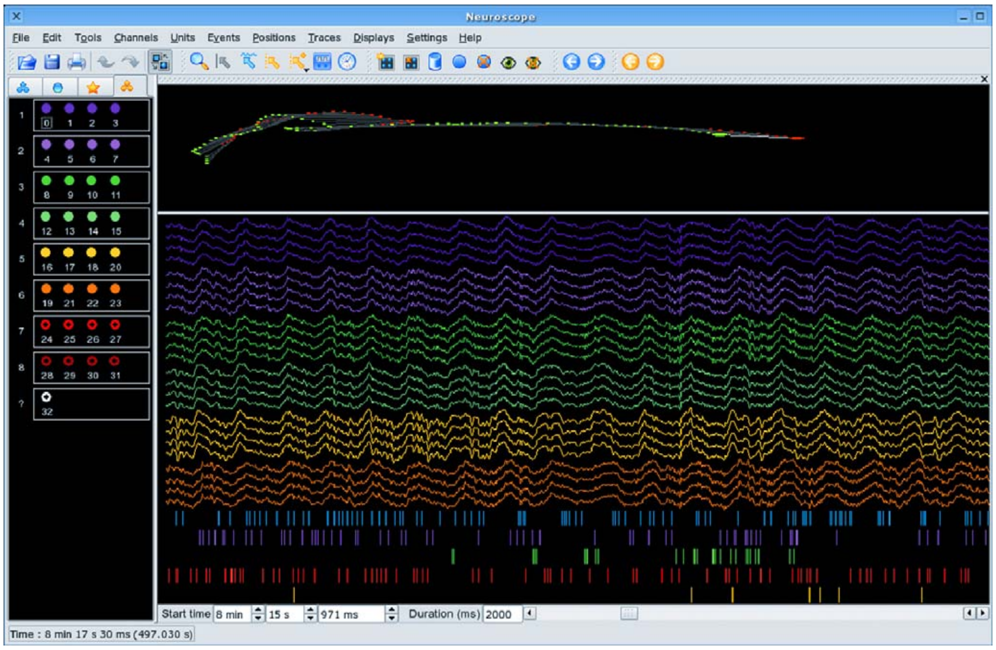

Neuroscope
File Edit Tools Channels Units Events Positions Traces Displays Settings Help
Start time 8 min 15 s 971 ms
Duration (ms) 2000
Time : 8 min 17 s 30 ms (497.030 s)

Fig. 6. NeuroScope. An overview, showing the trace view (electrophysiological signals, bottom view) and a position view (position tracking, here using two head lights, top view). For rapid identification, data from single sources (e.g., channel, cluster) are represented using the same color throughout the interface, including the selection palette (left panel). Several tools are shown in the tool bar (icons): zoom (magnifier lens), draw time line (grey arrow with vertical bar), select channels (blue arrow with horizontal trace), select event (yellow arrow with vertical dashed bar), add event (yellow arrow with vertical dashed bar and plus sign), measure (mutlimeter), select time (clock), previous and next spikes (blue arrows), previous and next events (yellow arrows), move channels to new group (square with four blue spheres and yellow star), remove channels from group (square with four blue spheres and red cross), discard channels (blue trash can), keep channels (blue sphere), skip channels (to label bad channels on silicon probes, crossed blue sphere), show channels (eye), hide channels (crossed eye). Trace start time and duration can be adjusted in the respective text fields.

set explicitly or interactively), for instance to show an overview of brain rhythms over longer periods of time, or help inspect spiking activity over shorter periods of time. A zooming tool is also provided to focus on even more restricted portions of the data (for instance, data sampled from a specific channel during a small time window).

5.2. Browsing unit activity

Unit activity can be represented as rasters below the continuous traces, vertical lines spanning the entire view, or full waveforms highlighted directly on the continuous traces (Fig. 8). Rasters allow for inspection of patterns of population activity, vertical lines emphasize relations between unit activity and local field events, and spike highlighting on the traces makes it possible to examine and compare waveforms at higher temporal resolution. For quick and convenient browsing, NeuroScope provides the possibility to directly move to the next or previous spike within a user-defined set of spike trains.

5.3. Position tracking

Provided the position of the animal was tracked during the experiment, NeuroScope can plot successive positions across time (using the same time window as for brain signals), each position being plotted as a connected set of colored points along a regular polygon, one point per head light carried by the animal (Fig. 6). By convention, the front light is plotted in red and all other lights in green. Thus, if the animal carries a single light, this will be represented as a red dot on the position tracking view, if it carries two lights (required to measure head direction), this will be represented as a segment with a red dot at the front end and a green dot at the back end, etc. To better estimate where the animal is located in the experimental apparatus, it is possible to display in the background an overhead photograph of the maze and/or the trajectory of the animal over the course of the entire experiment.

5.4. Browsing and editing events

Events can be plotted as vertical dashed lines in the trace view, and as cross marks in the position view, at the position occupied by the animal when they occurred (Fig. 6). Similar to spikes, one can directly move to the next or previous event in a user-defined set of events. Although NeuroScope is essentially a viewer, it also features a limited set of editing tools, allowing for addition, deletion and modification of events. This is particularly useful to manually indicate, or correct automatic detection of, field events such as hippocampal ripples, thalamo-cortical spindles, epileptic spikes, etc.

6. NDManager: a simple experimental parameter manager

Klusters and NeuroScope share a common parameter file (this is also used by other tools, as described in the next section). This file stores in XML format all the parameters required for data

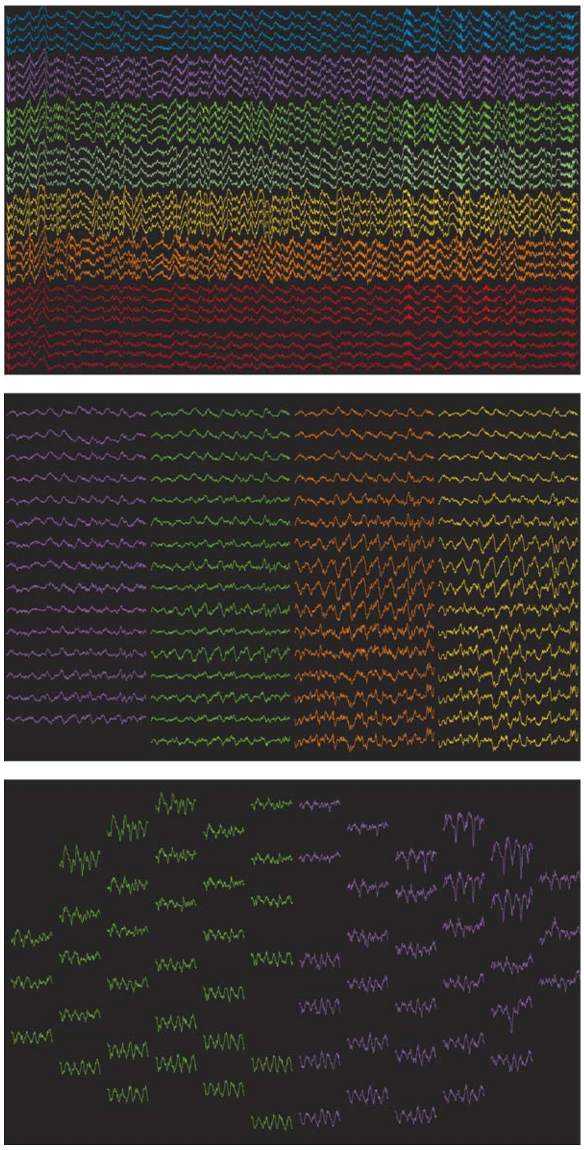


Fig. 7. Trace views: layouts. Electrode groups are displayed in different colors, either in vertical arrangements (top, tetrodes), in columnar arrangements (middle, silicon probes with regularly spaced recording sites), or in arbitrary layouts (bottom, epidural electrode array, arranged according to the topography of recording sites).

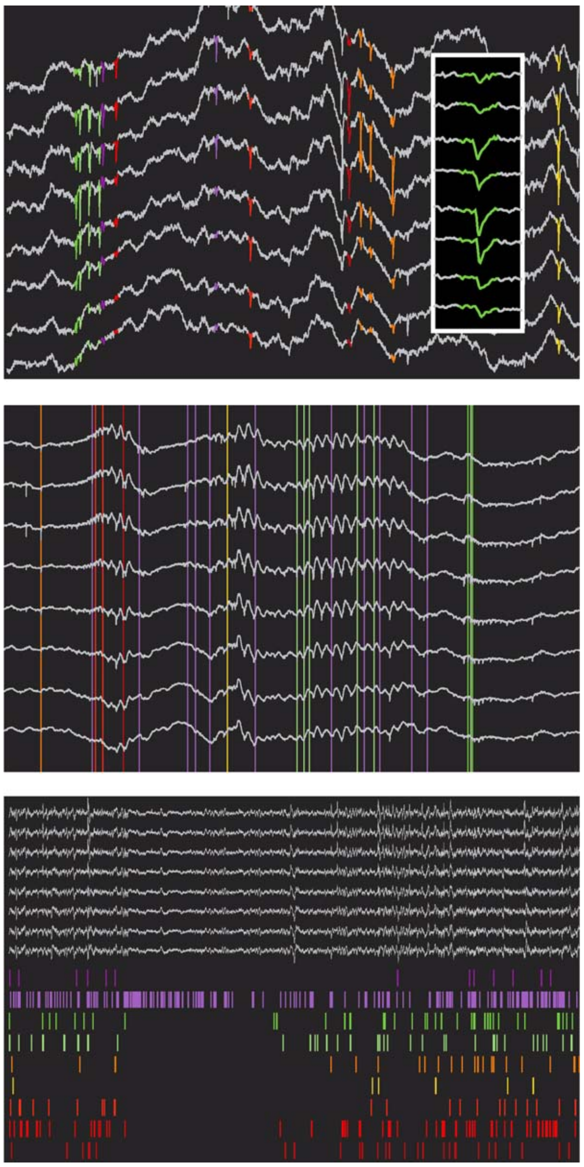


Fig. 8. Trace views: unit activity. Simultaneous display of clustered unit activity and field recordings. Unit activity can be shown as highlighted spike waveforms directly on the traces (top), emphasizing waveform shapes (inset, lower time scale), as vertical lines overlaid on the traces, emphasizing relations between unit and field activity (middle), or as rastergrams below the traces, emphasizing population firing patterns (bottom).

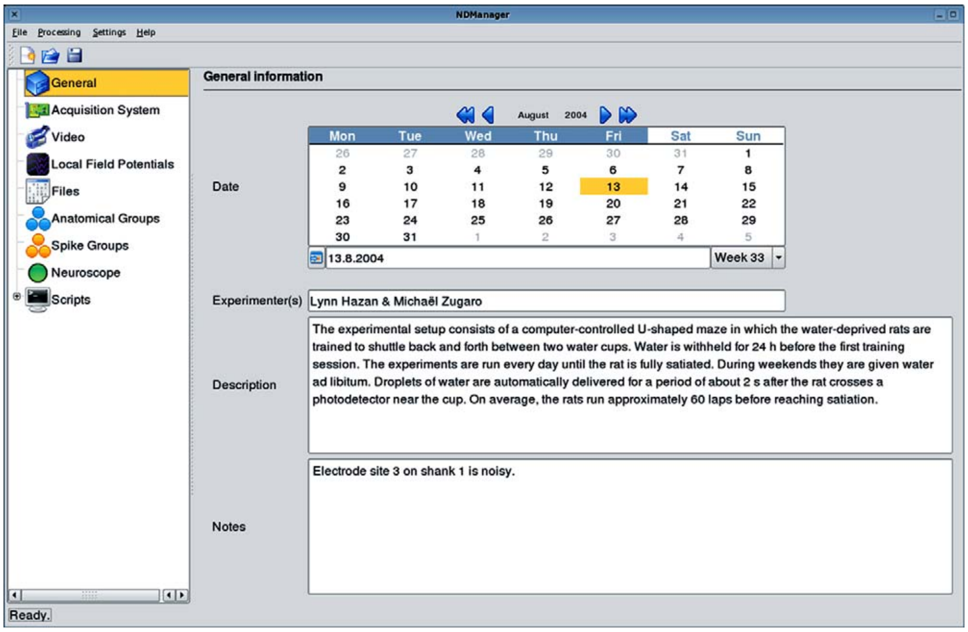
viewing and processing, including:

- general information: date, experimenter, description, notes;
- acquisition system: number of channels, sampling rate, resolution (in bits), offset, voltage range and amplification gain;
- video: sampling rate and frame size;
- file information: sampling rate and channel correspondence for processed files (e.g., low-pass filtered local field potentials, high-pass filtered data, average tetrode narrow-band field potentials, etc.)
- anatomical and spike groups: anatomical layout of the electrodes and groups for spike extraction and sorting;
- preprocessing parameters: software filter cutoff frequencies, spike extraction threshold, number of samples per waveform, number of principal components, etc.

All these parameters can be edited graphically using NDManager (Fig. 9). This simple application is designed to handle two related tasks: managing experimental parameters edition and data processing execution (as described below).

7. Integrated framework

Klusters, NeuroScope and NDManager are the three main components of a framework intended to provide all the necessary tools to process the data to bring it to suitable form for subsequent analysis: filtering, spike extraction and sorting, position tracking, event preprocessing, etc. Although several recording systems provide integrated tools to perform these steps, our



General information

Date: 13.8.2004 Week 33

Experimenter(s): Lynn Hazan & Michaël Zugaro

Description: The experimental setup consists of a computer-controlled U-shaped maze in which the water-deprived rats are trained to shuttle back and forth between two water cups. Water is withheld for 24 h before the first training session. The experiments are run every day until the rat is fully satiated. During weekends they are given water ad libitum. Droplets of water are automatically delivered for a period of about 2 s after the rat crosses a photodetector near the cup. On average, the rats run approximately 60 laps before reaching satiation.

Notes: Electrode site 3 on shank 1 is noisy.

Fig. 9. NDManager. This simple application is designed to handle the parameter file used Klusters and NeuroScope, as well as processing utilities. Parameters are grouped into categories (left panel) and include general information (right panel: date, experimenters, description of the experiment, notes), information about the acquisition system, preprocessing parameters, etc. NDManager is also designed to run preprocessing utilities. Thus, it constitutes a convenient central point from where parameters can be edited, files can be processed, and Klusters and NeuroScope can be started.

package offers a number of advantages: it allows researchers to modify and redistribute it as they see fit (hopefully bringing together a community of users and developers constantly improving this common resource); it includes powerful spike sorting capabilities by combining KlustaKwik, an efficient automatic clustering program, and Klusters, an advanced manual clustering application (most other packages only offer basic capabilities); it has the capability to eventually include all processing steps in an integrated suite.

In our framework, non-interactive processing such as high-pass filtering is typically done by command-line tools, usually written in C (but any other language can be used as well to develop additional custom components). Preprocessing does not require user intervention except for the definition of initial parameters (e.g., high-pass threshold). Although for technically sophisticated users, the command-line is a satisfactory user interface, for most users a simple graphical interface may be required, if only to define these parameters and start the actual processing more easily. This functionality is provided by NDManager, which has the capability to act as a graphical 'front-end' to any processing tools. This allows researchers to easily integrate their own processing tools within our framework without code modification. Only an intermediate script is required to read the parameters from the XML file (rather than from the command-line or legacy configuration files), and pass them along to the tools.

In practice, tools are executed via simple scripts (these can be written in sh, bash, perl, python, etc.), which first test the environment to anticipate possible errors (e.g., missing data files required for a specific task), then read parameters from the XML parameter file, and finally generate a command to start the respective tools. Along with each script and tool pair, a description file in XML format describes the parameters required for the task and provides a short help text. NDManager dynamically generates a graphical interface from this file, presenting the user a list of named parameters where values can easily be edited. Once all values have been provided, NDManager can execute the script, which reads these values and starts the tool accordingly.

8. Conclusion

The applications presented here allow for advanced processing and visualization of neurophysiological data sets including brain signals (action potentials and local field potentials), behavioral events and position tracking. These applications can be downloaded from the internet (http://neuroscope.sourceforge.net, http://klusters.sourceforge.net, and http://ndmanager.sourceforge.net). All the applications and tools described here are free software distributed under the GNU General Public License (GPL, see http://www.gnu.org/licenses/gpl.html). This license grants its users legal permission to copy, distribute and/or modify the software as they see fit. To this end, the source code is distributed alongside of the executable binaries. The only restriction imposed on the users is that should they decide to redistribute the software in its original or modified form, they must in turn grant their users the same rights. This guarantees that the software or any derivative of it will always remain freely available to the neuroscience community, and avoids the risk of vendor 'lock in' associated with proprietary software. We hope to attract an active community of users who could benefit from and contribute to this project.

Acknowledgements

We wish to thank Jozsef Csisvari, Anton Sirota, Lénaïc Monconduit, Judith Creso, David Robbe, Stephan Marguet, Sean Montgomery, Derek Buhl, Kenneth Harris, Peter Barthó and Hajime Hirase who helped define the feature set of the applications. Data used in the figures were provided by M. B. Z, Lénaïc Monconduit, Anton Sirota, David Robbe, Sean Montgomery and Peter Barthó. Supported by National Institutes of Health (NS34994, NS43157, MH54671) and the Human Frontier Science Foundation (to M. B. Z).

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at doi:10.1016/j.jneumeth.2006.01.017.

References

Abeles M, Goldstein MH. Multispike train analysis. In: Proceedings of the IEEE, vol. 65; 1977. pp. 762–73.

Barthó P, Hirase H, Monconduit L, Zugaro M, Harris KD, Buzsáki G. Characterization of neocortical principal cells and interneurons by network interactions and extracellular features. J Neurophysiol 2004;92:600–8.

Buzsáki G, Horváth Z, Urioste R, Hetke J, Wise K. High-frequency network oscillation in the hippocampus. Science 1992;256:1025–7.

Buzsáki G. Large-scale recording of neuronal ensembles. Nat Neurosci 2004;7:446–51.

Celeux G, Govaert G. A classification EM algorithm for clustering and two stochastic versions. Computational Statistics Data Anal 1992;14:315–32.

Csicsvari J, Hirase H, Czurko A, Buzsáki G. Reliability and state dependence of pyramidal cell-interneuron synapses in the hippocampus: an ensemble approach in the behaving rat. Neuron 1998;21:179–89.

Csicsvari J, Henze DA, Jamieson B, Harris KD, Sirota A, Barthó P, Wise KD, Buzsáki G. Massively parallel recording of unit and local field potentials with silicon-based electrodes. J Neurophysiol 2003;90: 1314–23.

Eichenbaum HB, Davis JL. Neuronal ensembles: strategies for recording and decoding. New York: Wiley-Liss; 2001.

Fee MS, Mitra PP, Kleinfeld D. Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-Gaussian variability. J Neurosci Methods 1996;69:175–88.

Hampson RE, Simeral JD, Deadwyler SA. Distribution of spatial and nonspatial information in dorsal hippocampus. Nature 1999;402:610–4.

Harris KD, Henze DA, Csicsvari J, Hirase H, Buzsáki G. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. J Neurophysiol 2000;84:401–14.

Hoffman KL, McNaughton BL. Coordinated reactivation of distributed memory traces in primate neocortex. Science 2002;297:2070–3.

Jutten C, Herault J. Blind separation of sources. Part I. An adaptive algorithm based on neuromimetic architecture. Signal Process 1991;24:1–10.

Khazipov R, Sirota A, Leinekugel X, Holmes GL, Ben-Ari Y, Buzsáki G. Early motor activity drives spindle bursts in the developing somatosensory cortex. Nature 2004;432:758–61.

McCormick DA, Connors BW, Lighthall JW, Prince DA. Comparative electrophysiology of pyramidal and sparsely spiny stellate neurons of the neocortex. J Neurophysiol 1985;54:782–806.

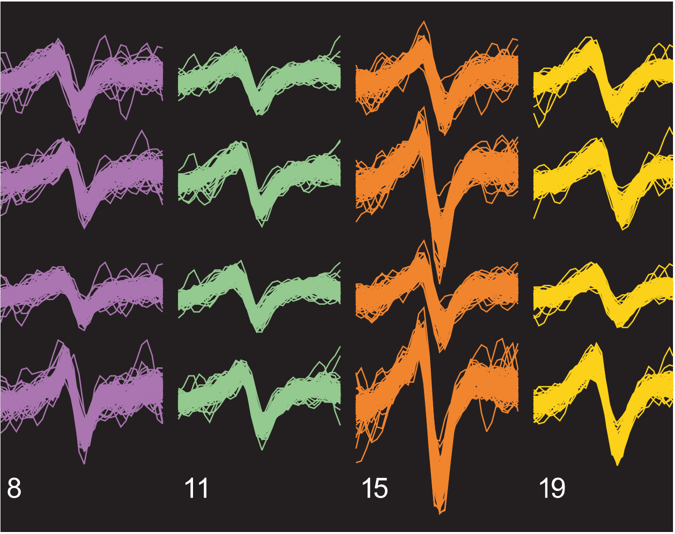
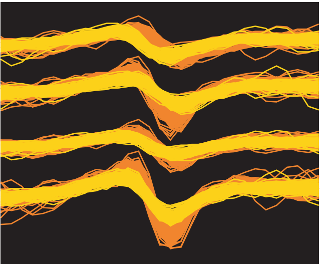
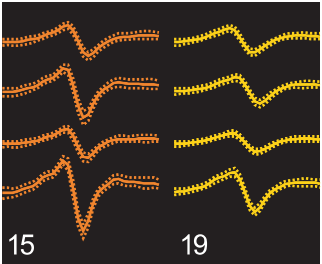
Nicolelis MA. Methods for neural ensemble recordings. Florida: CRC-Press; 1998.

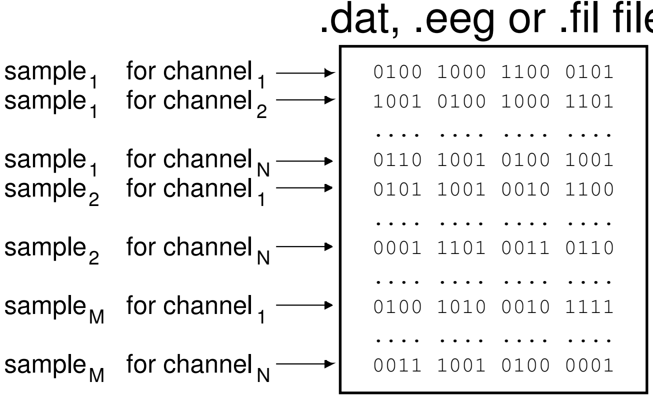
Nicolelis MA, Dimitrov D, Carmena JM, Crist R, Lehew G, Kralik JD, Wise SP. Chronic, multisite, multielectrode recordings in macaque monkeys. Proc Natl Acad Sci USA 2003;100:11041–6.

Wilson MA, McNaughton BL. Dynamics of the hippocampal ensemble code for space. Science 1993;261:1055–8.

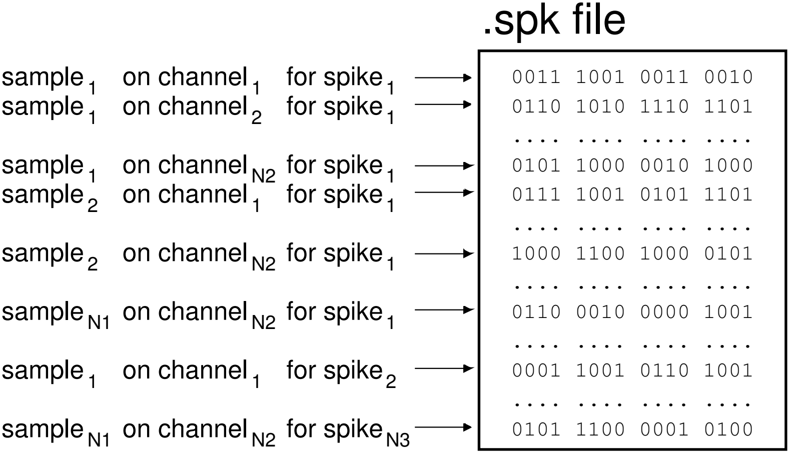
Wilson MA, McNaughton BL. Reactivation of hippocampal ensemble memories during sleep. Science 1994;265:676–9.

Zugaro MB, Monconduit L, Buzsáki G. Spike phase precession persists after transient intrahippocampal perturbation. Nat Neurosci 2005;8:67–71.

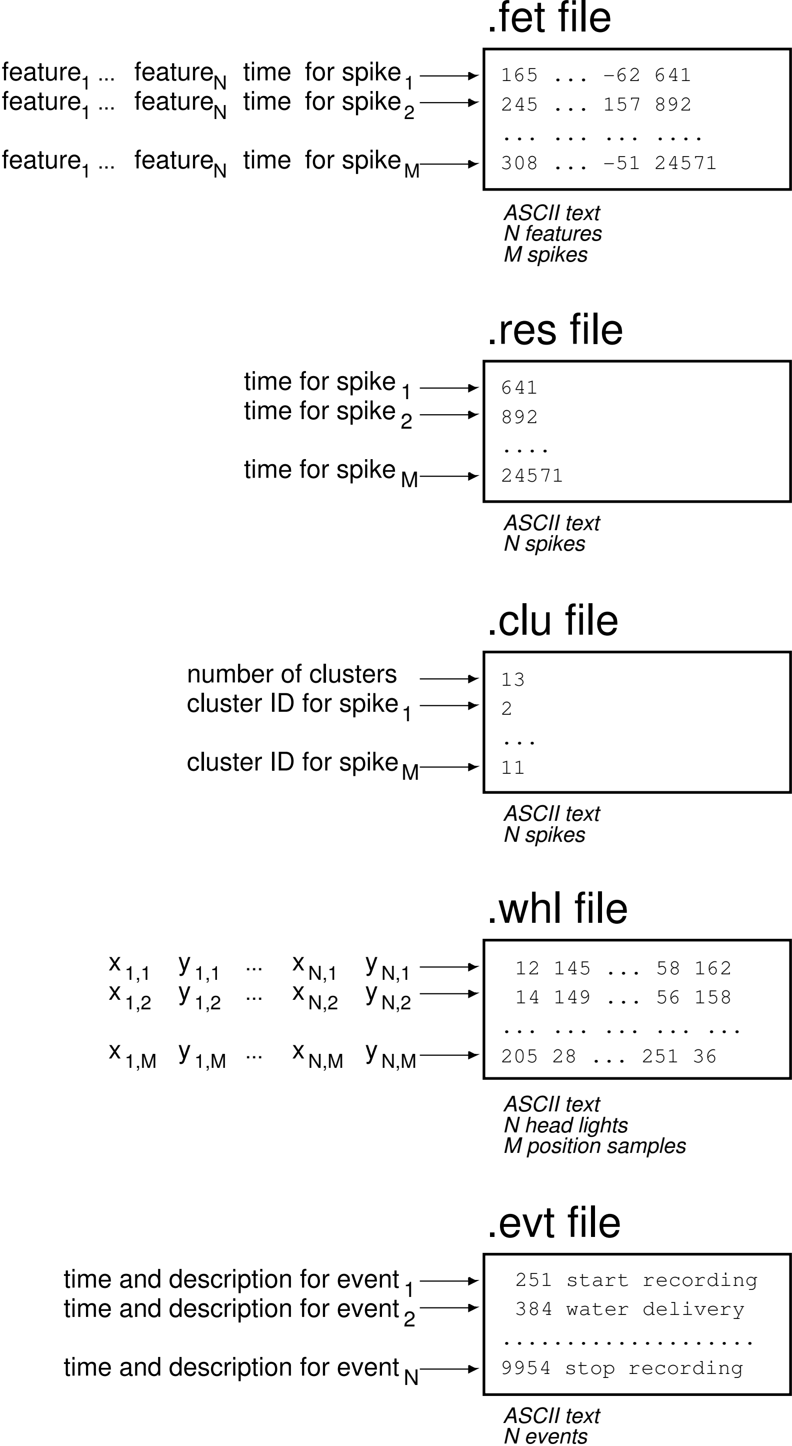

8
11
15
19


15
19


.dat, .eeg or .fil file
sample$_1$ for channel$_1$ → 0100 1000 1100 0101
sample$_1$ for channel$_2$ → 1001 0100 1000 1101
....
sample$_1$ for channel$_N$ → 0110 1001 0100 1001
sample$_2$ for channel$_1$ → 0101 1001 0010 1100
....
sample$_2$ for channel$_N$ → 0001 1101 0011 0110
....
sample$_M$ for channel$_1$ → 0100 1010 0010 1111
....
sample$_M$ for channel$_N$ → 0011 1001 0100 0001

16–bit binary (in this example)
N channels
M samples per channel


.spk file
sample$_1$ on channel$_1$ for spike$_1$ → 0011 1001 0011 0010
sample$_1$ on channel$_2$ for spike$_1$ → 0110 1010 1110 1101
....
sample$_1$ on channel$_{N2}$ for spike$_1$ → 0101 1000 0010 1000
sample$_2$ on channel$_1$ for spike$_1$ → 0111 1001 0101 1101
....
sample$_2$ on channel$_{N2}$ for spike$_1$ → 1000 1100 1000 0101
....
sample$_{N1}$ on channel$_{N2}$ for spike$_1$ → 0110 0010 0000 1001
....
sample$_1$ on channel$_1$ for spike$_2$ → 0001 1001 0110 1001
....
sample$_{N1}$ on channel$_{N2}$ for spike$_{N3}$ → 0101 1100 0001 0100

16–bit binary (in this example)
N1 samples per spike
N2 channels in group
N3 spikes



.fet file

feature$_1$ ... feature$_N$ time for spike$_1$
feature$_1$ ... feature$_N$ time for spike$_2$
feature$_1$ ... feature$_N$ time for spike$_M$

```
165 ... -62 641
245 ... 157 892
... ... ... ....
308 ... -51 24571
```

ASCII text
N features
M spikes

.res file

time for spike$_1$
time for spike$_2$
time for spike$_M$

```
641
892
....
24571
```

ASCII text
N spikes

.clu file

number of clusters
cluster ID for spike$_1$
cluster ID for spike$_M$

```
13
2
...
11
```

ASCII text
N spikes

.whl file

$x_{1,1}$ $y_{1,1}$ ... $x_{N,1}$ $y_{N,1}$
$x_{1,2}$ $y_{1,2}$ ... $x_{N,2}$ $y_{N,2}$
$x_{1,M}$ $y_{1,M}$ ... $x_{N,M}$ $y_{N,M}$

```
 12 145 ... 58 162
 14 149 ... 56 158
... ... ... ... ...
205 28 ... 251 36
```

ASCII text
N head lights
M position samples

.evt file

time and description for event$_1$
time and description for event$_2$
time and description for event$_N$

```
 251 start recording
 384 water delivery
....................
9954 stop recording
```

ASCII text
N events