14

Computational Intelligence in Electrophysiology: Trends and Open Problems

Cengiz Günay[1], Tomasz G. Smolinski[1], William W. Lytton[2], Thomas M. Morse[3], Padraig Gleeson[4], Sharon Crook[5], Volker Steuber[4,6], Angus Silver[4], Horatiu Voicu[7], Peter Andrews[8], Hemant Bokil[8], Hiren Maniar[8], Catherine Loader[9], Samar Mehta[10], David Kleinfeld[11], David Thomson[12], Partha P. Mitra[8], Gloster Aaron[13], and Jean-Marc Fellous[14]

[1] Dept. of Biology, Emory University, Atlanta, Georgia 30322, USA
[2] Depts of Physiology/Pharmacology and Neurology, State University of New York - Downstate, Brooklyn, New York 11203, USA
[3] Dept. of Neurobiology, Yale University, New Haven, CT 06510, USA
[4] Dept. of Physiology, University College London, London, UK
[5] Dept. of Mathematics and Statistics, Arizona State University, Tempe, Arizona, USA
[6] School of Computer Science, University of Hertfordshire, Hatfield, Herts, UK
[7] Dept. of Neurobiology and Anatomy, University of Texas Health Science Center, Houston, TX 77030, USA
[8] Cold Spring Harbor Laboratory, Cold Spring Harbor, NY 11724, USA
[9] Dept. of Statistics, University of Auckland, Auckland 1142, New Zealand
[10] School of Medicine, State University of New York - Downstate, Brooklyn, New York 11203, USA
[11] Dept. of Physics, University of California, San Diego, La Jolla, CA 92093, USA
[12] Dept. of Mathematics and Statistics, Queen's University, Kingston, ON, Canada
[13] Dept. of Biology, Wesleyan University, Middletown, CT 06459, USA
[14] Dept. of Psychology, University of Arizona, Tucson, AZ 85721, USA

Summary. This chapter constitutes mini-proceedings of the Workshop on Physiology Databases and Analysis Software that was a part of the Annual Computational Neuroscience Meeting CNS*2007 that took place in July 2007 in Toronto, Canada (`http://www.cnsorg.org`). The main aim of the workshop was to bring together researchers interested in developing and using automated analysis tools and database systems for electrophysiological data. Selected discussed topics, including the review of some current and potential applications of Computational Intelligence (CI) in electrophysiology, database and electrophysiological data exchange platforms, languages, and formats, as well as exemplary analysis problems, are presented in this chapter. The authors hope that the chapter will be useful not only to those already involved in the field of electrophysiology, but also to CI researchers, whose interest will be sparked by its contents.

14.1 Introduction

Recording and simulation in electrophysiology result in ever growing amounts of data, making it harder for conventional manual sorting and analysis methods to keep pace. The amount of electrophysiological data is increasing as more channels can be sampled and recording quality improves, while rapid advances in computing speed and capacity (e.g., in grid computing) have enabled researchers to generate massive amounts of simulation data in very short times. As a result, the need for automated analysis tools, with emphasis on Computational Intelligence-based techniques, and database systems has become widespread. The workshop on "Developing Databases and Analysis Software for Electrophysiology: Design, Application, and Visualization," organized by Cengiz Günay, Tomasz G. Smolinski, and William W. Lytton in conjunction with the 16th Annual Computational Neuroscience Meeting CNS*2007 (`http://www.cnsorg.org`), provided a venue for researchers interested in developing and using such tools to exchange knowledge and review currently available technologies and to discuss open problems.

This chapter constitutes mini-proceedings of the workshop and comprises of several selected contributions provided by the participants. In Section 14.2, Thomas M. Morse discusses the current uses and potential applications of CI for electrophysiological databases (EPDBs). Sections 14.3 by Padraig Gleeson et al., 14.4 by Horatiu Voicu, 14.5 by Cengiz Günay, and 14.6 by Peter Andrews et al., describe some currently available data-exchange and analysis platforms and implementations. Finally, Sections 14.7 by Gloster Aaron and 14.8 by Jean-Marc Fellous present some interesting open problems in electrophysiology with examples of analysis techniques, including CI-motivated approaches.

14.2 Computational Intelligence (CI) in electrophysiology: A review[1,2]

14.2.1 Introduction

There are 176 neuroscience databases listed in the Neuroscience Database Gateway [7]. Only one of these, Neurodatabase [32], currently has electrical recordings available, indicating that electrophysiology datasets are currently rarely publicly available. We review potential applications of electrophysiology databases (EPDBs) in hopes to motivate neuroscience, computer intelligence, computer science, and other investigators to collaborate to create and contribute datasets to EPDBs. We hope that some of these applications will

[1] Contribution by T.M. Morse.

[2] This work was supported in part by the NIH Grant 5P01DC004732-07 and 5R01NS011613-31.

inspire computational intelligence (CI) investigators to work on real (from future EPDBs) and simulated datasets that may then be immediately applicable to electrophysiology investigations and clinical applications. Currently, there are many tools available online [55] for the processing of fMRI data, ranging from simple statistical calculations to CI methods. There are many published tools available for the electrophysiologist (see below) but as yet there are no tool databases specifically for electrophysiologists (although the laboratory developing Neurodatabase also has plans to develop a tool database [33]), so we advocate tool database construction.

We describe the background for, and present open questions of EPDB CI tools, first in neuronal networks, then single cells, and finally ion channel/receptors recordings and analysis.

14.2.2 Neuronal Network recordings: Spike-Sorting

Spike sorting (the process of identifying and classifying spikes recorded in one or more electrodes as having been produced by particular neurons, by only using the recorded data) tools have been published since 1996 [27] and CI methods began to appear in 2002 [49]. Limited comparisons between tools have been made, for example recordings from an in vitro 2D network were used to compare wavelet packets decomposition with a popular principle components analysis method and an ordinary wavelet transform [49]. Spike sort method comparison studies have been limited by available data and by the subsequent availability of the data for further study.

The field needs a comprehensive review of spike sorting methods. Such a review would only be possible if many sets of data (from different laboratories) were available to use as the input for these methods. Different types of electrodes and preparations from different cell types and brain regions, species, etc. produce signals with different characteristics (neuronal population density, levels and types of activity, and levels and shape of noise). Having the traces available in publicly accessible EPDBs would then allow the methods to have their domains of applicability tested, as well as noting strengths and weaknesses of particular methods in particular domains.

The absence of publicly available extracellular recordings likely lead to the use of neural network model output to compare spike sorting routines, see for example [66]. ModelDB [47] is an additional source for network models, which if used as a resource for spike sorting tool comparison's could extend the testing with simulated epilepsy, trauma, and sleep activity [89, 101]. Comparison studies would be easier to create if spike-sorting tools were assembled into a tools database (as in for example those for MRI analysis [55]).

One important application of spike sorting methods are their anticipated role in neuroprosthetic devices. It has been shown that sorting spikes (which separates neurons that are being used for detection of neuronal patterns) increases the accuracy of reaching to a target prediction (a common task in neuroprosthetic research) between 3.6 and 6.4% [90].

It is a reasonable conjecture that spike sorting may also play a role in future understanding and/or treatment of Epilepsy [56]. One paper [87] references 10 off-line and only 3 online (real time) spike sorting technique papers and stresses the need for more research in online spike sorting methods. The authors of [87] had access to recordings from a chronic electrode implanted in the temporal lobe of an epileptic patient to test their methods.

The increasing size of recordings is another technical difficulty that electrophysiologists are facing. It is now possible to simultaneously record on the order of a hundred cells with multi-electrode arrays and on the order of a thousand cells with optical methods (up to 1300 cells in [53]). We suggest the online (real time) spike sorting algorithms are needed here to reduce the raw voltage or optical time series data to event times, thus vastly reducing the storage requirements for the data.

There is a pressing need for samples of all types of electrical recordings applicable to comparing spike-sorting methods to be deposited in a publicly accessible EPDB. If that data was available the following open questions in CI could be addressed. Does a single spike sorting algorithm outperform others in all recorded preparations?

If not which spike sorting methods work better in which preparations? Is there a substantial difference in thoroughness or accuracy between online (real time) and off-line (unrestricted processing time) spike sorting methods?

14.2.3 CI in single cell recordings

CI methods have been applied since the middle 1990's to single cell recordings to extract model parameters for models which describe the input output electrical function of the cell or channels or receptors within the cell (see for example the review in [102]).

Their appearance was a natural evolution in sophistication of early electrophysiology parameter extraction methods such as those in the famous Hodgkin Huxley (HH) 1952 paper [48]. In HH the authors use chemical species substitution and voltage clamp to isolate the voltage gated sodium and potassium channels. They created parameterized functions to describe the voltage and time dependence of the channels and extracted from their data the best fit for these functions.

Pharmacological isolation of additional channels, and morphological measurements guided by cable theory [81] enhanced the biological realism and types of channels and cells these HH-style models described. Passive parameter (membrane resistance, capacitance) extraction is commonly practiced, see for example [85, 94]. Pre-CI methods were provided by papers which used either brute force and conjugate descent methods [6] or transformations of gradient descent into a series of single parameter optimizations or a Chebyshev approximation [100].

Early CI methods in single cell model optimizations to electrophysiology traces are reviewed in [102]. This paper compared gradient descent, genetic algorithms, simulated annealing, and stochastic search methods. They found that simulated annealing was the best overall method for simple models with a small number of parameters, however genetic algorithms became equally effective for more complex models with larger numbers of parameters. A more recent paper incorporated simulated annealing search methods into an optimization of maximum conductances and calcium diffusion parameters with a hybrid fitness function and a revised (modified from the version in [102]) boundary condition handling method [104].

As the number of model parameters (the dimension of the parameter space) being determined increased, the complexity of the solution (the set of points in parameter space for which the error function is small enough or the fitness function large enough) also increased. Papers [39] have examined the parameter space that defines real cells or models by criteria of either a successful fit, or of the activity patterns of the cell [79], and have found that these spaces have non-intuitive properties due to the shapes of the solution spaces (see also Figure 2 of [6] and see also the call to investigate ways of reducing the parameter size and other issues in discussion in [102]). Tools to examine or to help describe these high dimensional model parameter spaces are important open CI research areas, because the (biological) cells parameters are traversing these spaces throughout the cells life. This is relevant to EPDBs because the model parameters extracted out of many collected recordings over different cell ages (embryonic, juvenile, adult) and environments would then be representative of the solution space of the cell, and hence the desire to view or to be able to describe that space.

Several of the previously cited papers use model current clamp data as targets for the search methods [104], single cell electrophysiology current clamp data [6], or both [102]. In addition, pre-CI paper methods which used model target data (for example [100]) could also be tested with electrophysiology data. The public availability of real training data would allow comparing the reliability and efficiency of model parameter extraction methods from electrophysiology data. The best extraction methods and associated training protocols would likely be determined per cell-type which would then determine which electrophysiology protocols would be performed and uploaded to the EPDBs in iterative improvements.

Voltage clamp caveat and another dataset

Voltage clamp protocols when performed in some cells exhibits what is called a space clamp error. The voltage clamp method attempts to maintain the cell at a voltage (a constant determined by the experimenter) and measures the (usually time-varying) injected current that is required to do so. The error arises when the voltage in parts of the cell that are distant from the electrode(s) have more influence from local currents than the electrode(s) due to either large local currents, small neuronal processes (reducing current flow from the electrode(s)), or being at a large distance from the electrode(s). In these cases membrane voltages drift from the command voltage and this is called "space-clamp error." A 2003 paper [91] was able to incorporate this phenomenon for measuring (the restricted case of) pharmacologically isolated hyperpolarizing channel densities, however their method is general enough in this domain to measure heterogeneous conductance density distributions (for those hyperpolarizing channels). We suggest that the currents recorded in [91, 92] would be helpful practice datasets if available in an EPDB for other electrophysiologists learning their method.

In addition to the above mentioned uses of EPDBs we present a speculative open question in CI; initial work could be done with models, however, it would only be through the availability of single cell recordings in EPDBs that the methods could be confirmed. Is it possible to characterize ion channels in (a well space clamped) experimental cell from a voltage clamp protocol that was the result of an optimized protocol developed and tested in models? Such a procedure would again only apply to well space-clamped cells. By "characterize" we mean to discover conductance densities, reversal potentials, and/or kinetics.

14.2.4 Single channels/receptors

Ion channel kinetic analysis preparations in experimental model cells (it is unfortunate that "model" has this ambiguity) such as HEK293, Xenopus egg, or isolated as single channels in excised membrane patches have had great success [88]. Voltage clamp errors are not a problem and the effects of superpositions of nearly similar or different multiple channel types can be eliminated. On the down side differences between native and experimental model cell intracellular and/or extracellular constituents, or the absence of these constituents in excised patches might make the native channel function differently than the isolated one. The results of the single channel studies are enormously useful because as it becomes known which channel genes are expressed (microarrays) at which densities in cells (staining methods or single channel population surveys) it will be possible to compare increasingly realistic models to nervous systems (see Figure 2 in [67]). What are optimal ways of deducing the kinetics of channels in single channel excised patch recordings?

Traditionally Markov modeling proceeds by choosing the number of states (nodes) and the transitions (connections) between them at the outset. The transition rates are then calculated or derived from experimental recordings (see [88] or more recently [20, 21, 41, 103]. Providing as many single channel recordings from each type of channel as possible would be invaluable for investigators and for developing automated Markov model construction tools. An open CI question: Is it feasible for optimization routines to search through different Markov model numbers of states and connections as a first step in a procedure that subsequently optimizes the transition rates, to find the simplest optimal model?

14.2.5 From today's EPDBs to tomorrow's

Investigators are currently using private EPDBs. We estimate, for example, there are about 110 papers to date in the Journal of Neurophysiology which report using their own EPDB locally to store neuronal recordings (145 papers total result from a free text search of "database neuron record" with about a 75% (actual EPDB) rate estimated from the first 37 papers). Investigators who are already using databases for their neuronal data might be able to use tools for making data publicly available like Nemesis [78] or Dan Gardner's group's method [30], or the commercial product Axiope [36] to upload datasets, after these tools evolve to interoperate with proposed EPDBs, or each other.

A common objection to EPDBs is that they would store large amounts of data that no one would use. We offer a different path by suggesting that EPDBs start out with just the samples of data that were suggested in this paper as immediately useful. Today's EP data is not too large.

If we estimated the average amount of electrophysiology data recorded for each paper at 10 GB then 100 papers worth is only one terabyte showing that the existing storage requirements are not as demanding as, for example, the fMRI data center faces. An open question is whether investigators will find this data useful for different reasons than for which it was created (e.g. useful for reasons not thought of yet); several groups are optimistic about the possibilities [16, 32].

14.2.6 Attributes of EPDBs

Every conceivable type of data in EPDBs is enumerated in the widely accepted Common Data Model [31]. In this framework we mention a couple of metadata preferences. The raw data could be stored along with (linked to) processing protocol instructions such as: we removed the artifacts from time steps t1 to t2 by zeroing, we filtered with a 4kHz filter, etc. Raw data is preferable because it permits the study of artifacts, noise, and the testing of filters. The explicit documentation of how the data was processed may be useful to other electrophysiologists. Standard processing methods could also be saved so that data could just point to the processing method rather than repeating it for each applicable dataset.

Measured channel densities stored in EPDBs would be highly valued by modelers. The published experimental conductance densities statistics (for example [38, 64, 91, 92] and references in [69, 70]) is infrequent and crucial for understanding synaptic integration and for making biologically realistic models. Channel densities provide modelers with the range of values that are realized in the biological system, further constraining the model.

14.2.7 Additional usefulness of EPDBs

The ability to quantitatively and/or graphically compare new models to experimental data that had been previously published and uploaded to EPDBs would be useful.

Comparing models to EP data have been done since at least HH, lending support to the model; see for example Figure 3 in [45] for a modified HH Na channel. Investigators must currently request traces from experimentalists, or retrieve experimental data with "data thief" or equivalent software [40, 51]. It would be nice to avoid that time and effort. See for example [18], where both experimental data and traces from calculations from experimental data were Data Thief'ed from 4 publications.

14.2.8 Conclusions

EPDBs could provide invaluable sources for comparisons between, and the development of new spike sorting tools and also single cell and ion channel/receptor electrophysiology methods and modeling. The authors hope that work in EPDBs will provide an exceptional example to this assessment from [5]: "Despite excitement about the Semantic Web, most of the world's data are locked in large data stores and are not published as an open Web of inter-referring resources."

We hope that EPDBs will become fertile members of the growing online population of neuroinformatics databases, fitting naturally among the connectivity, neuronal morphology, and modeling databases to support the study of the electrical functioning of the nervous system and excitable membranes.

14.3 Using NeuroML and neuroConstruct to build neuronal network models for multiple simulators[3,4]

14.3.1 Introduction

Computational models based on detailed neuroanatomical and electrophysiological data have been used for many years to help our understanding of the function of the nervous system. Unfortunately there has not been very widespread use of such models in the experimental neuroscience community. Even between computational neuroscientists, there are issues of compatibility of published models, which have been created using a variety of simulators and programming languages. Here we discuss new standards for specifying such models and a graphical application to facilitate the development of complex network models on multiple simulators.

[3] Contribution by P. Gleeson, S. Crook, V. Steuber, R.A. Silver.

[4] This work has been funded by the MRC and the Wellcome Trust.

14.3.2 NeuroML (`http://www.neuroml.org`)

The Neural Open Markup Language project, NeuroML [22, 35], is an international, collaborative initiative to create standards for the description and interchange of models of neuronal systems. The need for standards that allow for greater software interoperability is driving the current NeuroML standards project, which focuses on the key objects that need to be exchanged among existing applications and tries to anticipate those needed by future neuroscience applications.

The current standards are arranged in Levels (Figure 14.1), with each subsequent Level increasing the scope of the specification. Level 1 of the standards provides both a framework for describing the metadata associated with any neuronal model (e.g. authorship, generic properties, comments, citations, etc.) and allows specifications of neuroanatomical data, e.g. the branching structure of neurons, histological features, etc. Morphological data from various sources, e.g. Neurolucida reconstructions, can be converted into this format, termed MorphML [22, 80] for reuse in compartmental modeling simulators, etc.

Level 2 allows the specification of models of conductance based multicompartmental neurons. Inhomogeneous distributions of membrane conductances, subcellular mechanisms and passive cellular properties can be described for cells based on MorphML. Models of voltage and ligand gated ion channels and synaptic mechanisms can be described with ChannelML. Level 3 of the specification is aimed at network models. Populations of neurons in 3D can be defined by providing an explicit list of all neuronal locations, or by providing an implicit enumeration (e.g. a grid arrangement or a random arrangement). Similarly, connectivity can be specified using an explicit list of connections or implicitly by giving an algorithm for defining connectivity rules, cell location specificity of synaptic types, etc.

The advantage of using XML for the descriptions is that files can be checked for completeness against a published standard, i.e. any missing fields in a model description can be automatically detected. Another advantage is that XML files can easily be transformed into other formats. There are currently mappings to the GENESIS [11] and NEURON [46] simulators.

The latest version of the NeuroML specifications can be found online at `http://www.morphml.org:8080/NeuroMLValidator` along with example files at each of the Levels described. There is also the possibility of validating NeuroML files to ensure their compliance to the current version of the standards. The NeuroML project is working closely with a number of application developers to ensure wider acceptance of the standard.

14.3.3 neuroConstruct

One application which uses the NeuroML standards to facilitate model development is neuroConstruct [34]. This is a platform independent software tool

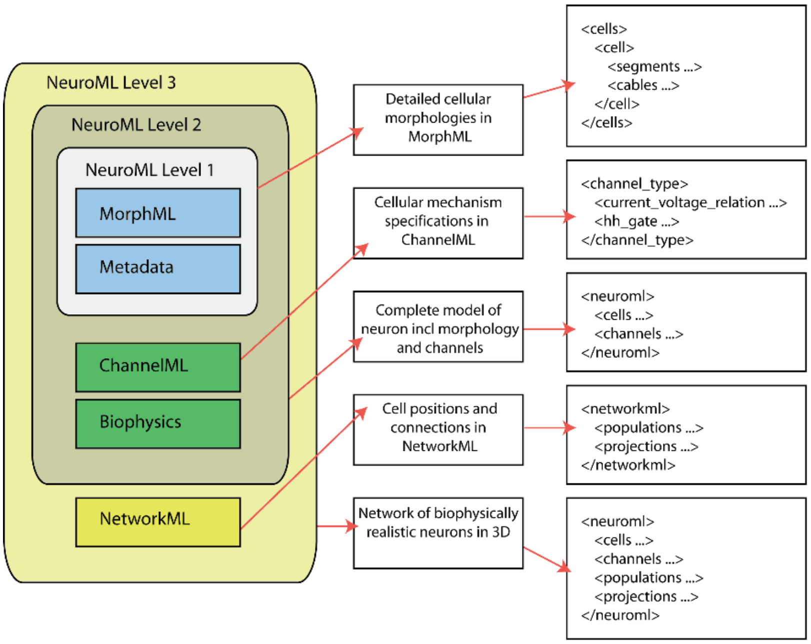

NeuroML Level 3
NeuroML Level 2
NeuroML Level 1
MorphML
Metadata
ChannelML
Biophysics
NetworkML
Detailed cellular morphologies in MorphML
Cellular mechanism specifications in ChannelML
Complete model of neuron incl morphology and channels
Cell positions and connections in NetworkML
Network of biophysically realistic neurons in 3D
<cells> <cell> <segments ...> <cables ...> </cell> </cells>
<channel_type> <current_voltage_relation ...> <hh_gate ...> </channel_type>
<neuroml> <cells ...> <channels ...> </neuroml>
<networkml> <populations ...> <projections ...> </networkml>
<neuroml> <cells ...> <channels ...> <populations ...> <projections ...> </neuroml>

Fig. 14.1. The 3 Levels in the current NeuroML specifications.

for constructing, visualizing and analyzing conductance-based neural network models with properties that closely match the 3D neuronal morphology and connectivity of different brain regions (Figure 14.2). A user friendly GUI allows models to be built, modified and run without the need for specialist programming knowledge, providing increased accessibility to both experimentalists and theoreticians studying network function. neuroConstruct generates script files for NEURON or GENESIS which carry out the numerical integration.

Networks of cells with complex interconnectivity can be created with neuroConstruct, simulations run with one or both of the supported simulators and the network behavior analyzed with a number of inbuilt tools in neuroConstruct. neuroConstruct is freely available from `http://www.neuroConstruct.org`.

14.3.4 Conclusions

The creation of more detailed models of neuronal function will require the interaction of a range of investigators from many backgrounds. Having a common framework to describe experimental findings and theoretical models about function will greatly aid these collaborations. Appropriate tools for creating models and testing ideas are also needed in this process. The ongoing work on NeuroML and neuroConstruct can help towards these goals.

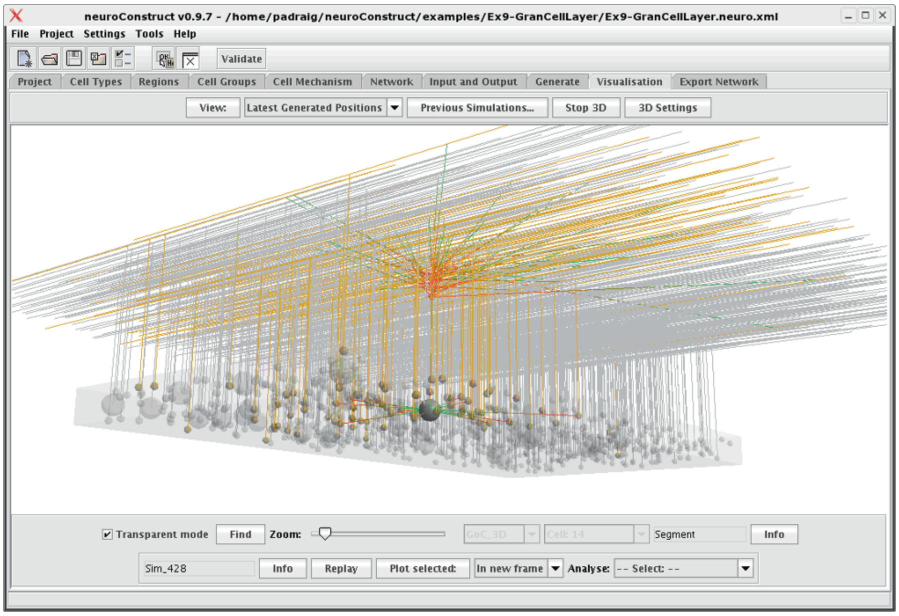

neuroConstruct v0.9.7 - /home/padraig/neuroConstruct/examples/Ex9-GranCellLayer/Ex9-GranCellLayer.neuro.xml
File Project Settings Tools Help
Validate
Project Cell Types Regions Cell Groups Cell Mechanism Network Input and Output Generate Visualisation Export Network
View: Latest Generated Positions Previous Simulations... Stop 3D 3D Settings
Transparent mode Find Zoom: GoC_3D Cell: 14 Segment Info
Sim_428 Info Replay Plot selected: In new frame Analyse: -- Select: --

Fig. 14.2. Screenshot of a network model of the granule cell layer of the cerebellum created in neuroConstruct.

14.4 Time saving technique for developing and maintaining user interfaces[5]

Developing computer analyses and simulations for experimental data in neuroscience often requires interactive access for most of the variables used in the computer applications. Most of the time, this interaction is accomplished by developing a user interface that needs to be maintained and updated as the application evolves. This section presents a simple approach that avoids the development of a user interface, yet allows the user full functionality in manipulating the application. Although this project is targeted for the Windows operating system, the same strategy can be used with other operating systems.

The gist of the approach is to allow the application to modify internal variables by receiving real-time commands through the messaging system provided by the operating system. To achieve this goal the following tasks need to be completed: (1) inside the application, build a map between the name of the variables that represent biophysical measures and the actual variables that represent them, (2) implement a function that receives commands from other applications (3) build a small interpreter that can take commands of the form ‘Membrane_voltage -80’ meaning update the Membrane_voltage variable

[5] Contribution by H. Voicu.

with the value -80, and (4) build a small editor that can send commands to other applications through the messaging system of the operating system. The complete project can be downloaded from: `www.voicu.us/software.zip`.

The first task is the most straightforward and can be easily implemented in C++ as shown below:

```
float EL_Ca3_pyr =
  make_float("ca3_pyr_cells.var",&EL_Ca3_pyr,"EL_Ca3_pyr",-60);

float make_float(char *fn,float *p, char *s, float val) {
if (u_i_float_var_counter >= NUM_MAX_OF_UI_FLOAT_VARS)
  return(val);

u_i_var_float_ptr_array[u_i_float_var_counter] = p;
strcpy(u_i_var_float_str_array[u_i_float_var_counter],s);
strcpy(u_i_var_float_file_name_array[u_i_float_var_counter],fn);
u_i_float_var_counter++; return(val);
}
```

Let us assume that `EL_Ca3_pyr` represents the resting potential of pyramidal cells in the CA3 subfield of the hippocampus. This variable is initialized with -60. The function `make_float` builds a map between the location in memory where the value of `EL_Ca3_pyr` is stored and the string "`EL_Ca3_pyr.`" The string "`ca3_pyr_cells.var`" defines the name of the file in which `EL_Ca3_pyr` will be listed with its initial value when the program starts. This is handled by a separate function `generate_list_of_variables` which is included in the source code.

Since the messages we plan to send are short we can use the actual content of the pointers `WPARAM wp`, `LPARAM lp` of the function SendMessage to contain the information. This feature makes the application compatible with the GWD editor which we use for sending commands. The function that receives the command must have two important features. It must be able to concatenate partial messages and preserve their order. Preserving temporal order is particularly important since the repeated update of a variable can make messages corresponding to different commands arrive about the same time. To discriminate between partial messages that belong to different commands, each partial message is prefixed by a byte representing its identity. The function `OnMyMessage` takes the partial message contained in the 4 byte long variables wp and lp and concatenates it to the current command. When the full command is received, it is processed using the `Process_Message_From_User function`.

```
LRESULT CMainFrame::OnMyMessage(WPARAM wp, LPARAM lp) {
rm_multiplexer = (unsigned char)(wp & 0xff); wp >>= 8;
if (bool_receiving_message_from_user[rm_multiplexer]) {
for (int i1 =0; i1<3; i1++, rec_msg_cntr[rm_multiplexer]++, wp >>= 8) {
rec_message[rm_multiplexer][rec_msg_cntr[rm_multiplexer]] = wp & 0xff;
if (rec_message[rm_multiplexer][rec_msg_cntr[rm_multiplexer]]==0)
  goto exec_comm;
}
for (int i1 =0; i1<4; i1++, rec_msg_cntr[rm_multiplexer]++, lp >>= 8) {
rec_message[rm_multiplexer][rec_msg_cntr[rm_multiplexer]] = lp & 0xff;
if (rec_message[rm_multiplexer][rec_msg_cntr[rm_multiplexer]]==0)
  goto exec_comm;
}} else {
rec_msg_cntr[rm_multiplexer] = 0;

for (int i1 = 0 ; i1<3 ; i1++, rec_msg_cntr[rm_multiplexer]++, wp >>= 8) {
rec_message[rm_multiplexer][rec_msg_cntr[rm_multiplexer]] = wp & 0xff;
if (rec_message[rm_multiplexer][rec_msg_cntr[rm_multiplexer]]==0)
  goto exec_comm;
}
for (int i1 =0 ;i1<4; i1++, rec_msg_cntr[rm_multiplexer]++, lp >>= 8) {
rec_message[rm_multiplexer][rec_msg_cntr[rm_multiplexer]] = lp & 0xff;
if (rec_message[rm_multiplexer][rec_msg_cntr[rm_multiplexer]]==0)
  goto exec_comm;
}
bool_receiving_message_from_user[rm_multiplexer] = 1;
}
return 0;

exec_comm:
bool_receiving_message_from_user[rm_multiplexer] = 0;
Process_Message_From_User(rec_message[rm_multiplexer]);
return 0;
}
```

The `Process_Message_From_User` function is a standard interpreter for the commands received by the application. It uses the mapping from variable names to variable content build by the `make_float` function to update variables the same way a user interface does. The same function can be used to interpret commands stored in a file provided in the command line.

The most involved task in this project is the development of a supporting application that can send commands to the analysis or simulation application. Although rather lengthy, this task can be accomplished in a straightforward way. The alternative is to use an application that already has this capability, like the GWD text editor. Another important feature of the supporting application is the ability of increasing and decreasing the values of the variables declared with `make_float`. This ability can be easily implemented in the GWD editor with the help of macros. The source code shows how the value of a variable is increased/decreased by a value determined by the position of the cursor with respect to the decimal point and sent to the application.

A typical "*.var" file looks like this:

```
mfc_hippocampus

gL_Ca3_pyr 0.004000
gL_Ca3_pyr 0.004000
min_max 0.001000 0.009000
```

The first line contains the name of the application where the commands are sent. Each variable appears in two consecutive lines. The macros in the GWD program assume that the second line stores the default value. If the two consecutive lines are followed by a line containing the 'min_max' keyword than the macros do not change the variable below or above the values specified on that line.

There are two important advantages of this technique. First, the introduction of a new variable does not require adding code to the application except the declaration itself. Second, a large number of variables can be organized in different "*.var" files avoiding clutter and the development of additional GUI windows.

14.5 PANDORA's Toolbox: database-supported analysis and visualization software for electrophysiological data from simulated or recorded neurons with Matlab[6]

As the amount of data for EPDBs increase, its organization and labeling becomes more difficult. It is critical to reduce this difficulty since the analysis of the results depends on the accurate labeling of the data. For recordings, the difficulty comes from associating recorded data traces with multiple trials, different recording approaches (e.g., intracellular vs. extracellular), stimulation protocol parameters, the time of recording, the animal's condition, the drugs used, and so on [3, 17]. For simulations, the difficulty comes from the large number of possible simulation parameter combinations of ion channel densities, channel kinetic parameters, concentration and flux rates. The difficulty remains high irrespective of the actual model used, be it a neuron model containing multiple types of Hodgkin-Huxley ion channels [6, 96, 102] or a neuron model with Markov channels [86] and detailed morphological reconstructions, or a model of molecular pathways and processes [95]. Although simpler versions of these models are employed in network simulations, the number of parameters are still large to account for variances in neuron distributions, variable connection weights, modulators, time constants, and delays [50, 58, 97].

Although accurate labeling of data is critical, often custom formats and tools are used for data storage and analysis. These formats range from keeping

[6] Contribution by C. Günay.

data in human-readable text files to proprietary or ad-hoc binary formats. It is rare that a proper database management system is employed. Using database management systems has the advantage of automatically labeling the data. This "metadata," which is created when the experimental data is inserted into a database, remains with the data during different analysis steps and it may reach the final plots. Even though for small datasets, the consistency of the experimental parameters can be maintained by manual procedures, having automated systems that keep track of data becomes invaluable for larger datasets. Furthermore, a database system provides formal ways to manage, label and query datasets, and it maintains relationships between dataset elements.

The question is, then, to find the most optimal type of database system for storing and analyzing electrophysiological data. Storing experiment or simulation parameters in a database is simpler compared with storing the raw outputs of the experiments (e.g., voltage traces). Preserving the raw recorded data is essential for analyzing and understanding the results of an experiment. Especially with large datasets, with several hundred gigabytes (billion bytes), it becomes difficult to store, search and process the raw data quickly and efficiently for answering specific questions. But questions can be answered much faster if features that pertain to the possible questions in hand are extracted and placed in a more compact database format. Then, once interesting entries are found from the features and parameters in the database, the raw data can be consulted again for validation, visualization and further analysis. This database of parameters and extracted features can be subject to several types of numerical and statistical analyses to produce higher-level results.

The Neural Query System (NQS) [62] provided such a database system for analyzing results of neural simulations. NQS is a tool integrated into the Neuron simulator [46] to manage simulations and record their features in a database for further analysis. Here, we introduce PANDORA's toolbox[7] that provides this type of database support for both simulated and recorded data. It currently offers offline analysis within the Matlab environment for intracellular neural recordings and simulations in current-clamp mode, stimulated with a current-injection protocol. PANDORA provides functions to extract important features from electrophysiology data such as spike times, spike shape information, and other special measurements such as rate changes; after the database is constructed, second tier numerical and statistical analyses can be performed; and both raw data and other intermediate results can be visualized.

PANDORA was designed with flexibility in mind and we present it as a tool available for other electrophysiology projects. PANDORA takes a simplified approach providing a native Matlab database that has the advantage of being independent of external applications (although it can communicate

[7] PANDORA stands for "Plotting and Analysis for Neural Database-Oriented Research Applications."

with them) and thus requires no additional programming in a database language. However, it inherits limitations of the Matlab environment in terms of being not as well optimized for speed and memory usage. In particular, a database table to be queried must completely fit in the computer's memory. PANDORA is distributed with an Academic Free License and can be freely downloaded from `http://senselab.med.yale.edu/SimToolDB`.

14.6 Chronux: A Platform for Analyzing Neural Signals[8,9]

14.6.1 Introduction

Neuroscientists are increasingly gathering large time series data sets in the form of multichannel electrophysiological recordings, EEG, MEG, fMRI and optical image time series. The availability of such data has brought with it new challenges for analysis, and has created a pressing need for the development of software tools for storing and analyzing neural signals. In fact, while sophisticated methods for analyzing multichannel time series have been developed over the past several decades in statistics and signal processing, the lack of a unified, user-friendly, platform that implements these methods is a critical bottleneck in mining large neuroscientific datasets.

Chronux is an open source software platform that aims to fill this lacuna by providing a comprehensive software platform for the analysis of neural signals. It is a collaborative research effort currently based at Cold Spring Harbor Laboratory that draws on a number of previous research projects [8–10, 27, 60, 71, 77, 98]. The current version of Chronux includes a Matlab toolbox for signal processing of neural time series data, several specialized mini-packages for spike sorting, local regression, audio segmentation and other tasks. The eventual aim is to provide domain specific user interfaces (UIs) for each experimental modality, along with corresponding data management tools. In particular, we expect Chronux to grow to support analysis of time series data from most of the standard data acquisition modalities in use in neuroscience. We also expect it to grow in the types of analyses it implements.

14.6.2 Website and Installation

The Chronux website at `http://chronux.org/` is the central location for information about the current and all previous releases of Chronux. The home page contains links to pages for downloads, people, recent news, tutorials,

[8] Contribution by P. Andrews, H. Bokil, H. Maniar, C. Loader, S. Mehta, D. Kleinfeld, D. Thomson, P.P. Mitra.

[9] This work has been supported by grant R01MH071744 from the NIH to P.P. Mitra.

various files, documentation and our discussion forum. Most of the code is written in the Matlab scripting language, with some exceptions as compiled C code integrated using Matlab mex functions. Chronux has been tested and runs under Matlab releases R13 to the current R2007a under the Windows, Macintosh and Linux operating systems. Extensive online and within-Matlab help is available.

As an open source project released under the GNU Public License GPL v2, we welcome development, code contributions, bug reports, and discussion from the community. To date, Chronux has been downloaded over 2500 times. Questions or comments about can be posted on our discussion forum at `http://chronux.org/forum/` (after account registration). Announcements are made through the Google group chronux-announce.

14.6.3 Examples

This section contains examples of Chronux usage, selected to show how it can handle several common situations in analysis of neural signals. We focus on spectral analysis, and local regression and likelihood since these techniques have a wide range of utility.

Spectral Analysis

The spectral analysis toolbox in Chronux is equipped to process continuous valued data and point process data. The point process data may be a sequence of values, such as times of occurrence of spikes, or a sequence of counts in successive time bins. Chronux provides routines to estimate time-frequency spectrograms and spectral derivatives, as well as measures of association such as cross-spectra and coherences. Univariate and multivariate signals may be analyzed as appropriate. Where possible, Chronux provides confidence intervals on estimated quantities using both asymptotic formulae based on appropriate probabilistic models, as well as nonparametric bands based on the Jackknife method. Finally, Chronux includes various statistical tests such as the two-group tests for the spectrum and coherence, nonstationarity test based on quadratic inverse theory, and the F-test to detect periodic signals in a colored background. The latter is particularly useful for removing 50 or 60 Hz line noise from recorded neural data.

Space constraints preclude covering all of spectral analysis here, but the functions generally have a uniform function calling signature. We illustrate three canonical routines below.

mtspectrumc

As a first example, we show how to estimate the spectrum of a single trial local field potential measured from macaque during a working memory task. Figure 14.3 shows the spectrum estimated by the Chronux function

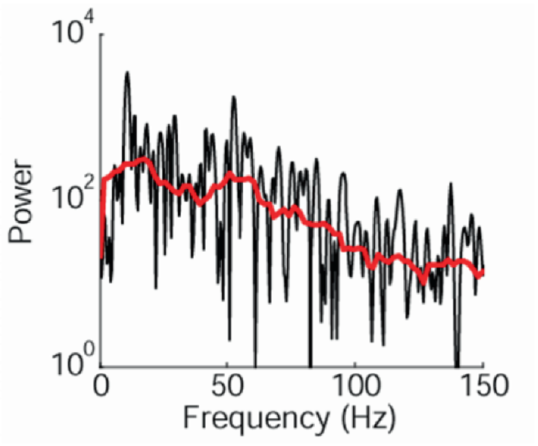


Fig. 14.3. Comparison of a periodogram (black) and multitaper estimate (red) of a single trial local field potential measurement from macaque during a working memory task. This estimate used 9 tapers.

`mtspectrumc`. For comparison we also display the ordinary periodogram. In this case, `mtspectrumc` was called with params.tapers=[5 9].

The Matlab calling signature of the `mtspectrumc` function is as follows:

```
[S,f,Serr] = mtspectrumc( data, params );
```

The first argument is the data matrix in the form of times $\times$ trials or channels, while the second argument *params* is a Matlab structure defining the sampling frequency,[10] the time-bandwidth product used to compute the tapers, and the amount of zero padding to use. It also contains flags controlling the averaging over trials and the error computation. In this example, params.tapers was set to be [5 9], thus giving an estimate with a time bandwidth product 5, using 9 tapers (For more details on this argument, see below).

The three variables returned by `mtspectrumc` are, in order, the estimated spectrum, the frequencies of estimation, and the confidence bands. The spectrum is in general two-dimensional, with the first dimension being the power as a function of frequency and the second dimension being the trial or channel. The second dimension is 1 when the user requests a spectrum that is averaged over the trials or channels. The confidence bands are provided as a lower and upper confidence band at a p value set by the user. As indicated by the last letter c in its name, the routine `mtspectrumc` is applicable to continuous valued data such as the local field potential or the EEG. The corresponding routines for point processes are `mtspectrumpt` and `mtspectrumpb`, applicable

[10] The current version of Chronux assumes continuous valued data to be uniformly sampled

to point processes stored as a sequence of times and binned point processes, respectively.

mtspecgramc

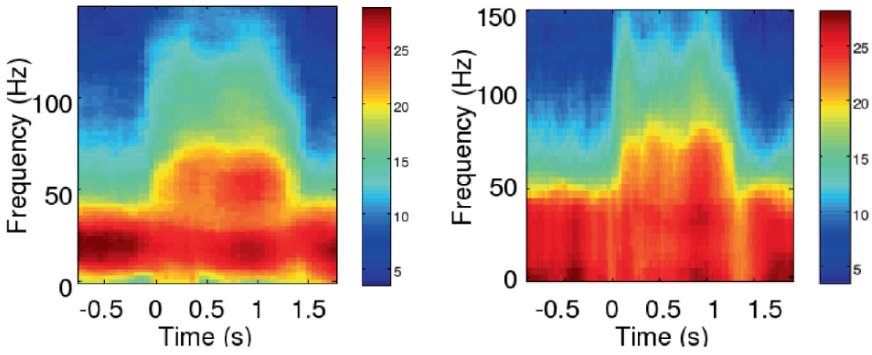


Fig. 14.4. The effect of changing the components of the time-bandwidth product TW. a) T = 0.5s, W = 10Hz. b) T = 0.2s, W = 25Hz. Data from macaque monkey performing a working memory task. Sharp enhancement in high frequency power occurs during the memory period.

The second example is a moving window version of `mtspectrumc` called `mtspecgramc`. This function, and `mtspecgrampt` and `mtspecgrampb`, calculate the multitaper spectrum over a moving window with user adjustable time width and step size. The calling signature of this function is:

```
[S,t,f,Serr] = mtspecgramc( data, movingwin, params );
```

Note that the only differences from the `mtspectrumc` function signature are in the function name, the additional `movingwin` argument and the addition of a return value `t` which contains the centers of the moving windows. The `movingwin` argument is given as `[winsize winstep]` in units consistent with the sampling frequency. The returned spectrum here is in general three dimensional: times $\times$ frequency $\times$ channel or trial.

The variable params.tapers controls the computation of the tapers used in the multitaper estimate. params.tapers is a two-dimensional vector whose first element, TW, is the time-bandwidth product, where T is the duration and W is the desired bandwidth. The second element of params.tapers is the number of tapers to be used. For a given TW, the latter can be at most $2TW - 1$. Higher order taper functions will not be sufficiently concentrated in frequency and may lead to increased broadband bias if used.

Figure 14.4 shows the effect on the spectrogram of changing the time-bandwidth product. The data again consists of local field potentials recorded from macaque during a working memory task.

coherencycpt

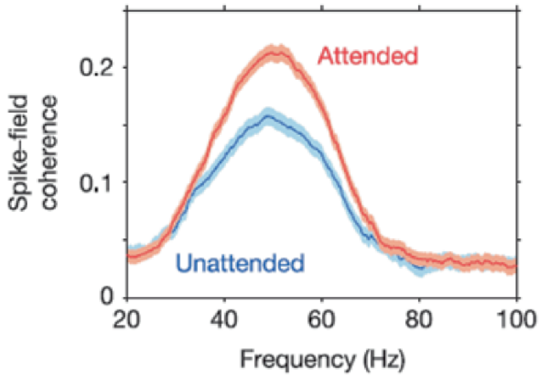


Fig. 14.5. The spike-field coherence recorded from visual cortex of monkey, showing significant differences between the attended and unattended conditions. In addition to the coherence in the two conditions, we also show the 95% confidence bands computed using the Jackknife.

Figure 14.5 [105] shows significant differences in the spike-field coherence recorded from the primary visual cortex of monkeys during an attention modulation task. The coherences were computed using `coherencycpt`. This function is called with two timeseries as arguments: the continuous LFP data and the corresponding spikes which are stored as event times. It returns not only the magnitude and phase of the coherency, but the cross-spectrum and individual spectra from which the coherence is computed. As with the spectra, confidence intervals on the magnitude and phase of the coherency may also be obtained.

Locfit

The Locfit package by Catherine Loader [61] is included in Chronux. Locfit can be used for local regression, local likelihood estimation, local smoothing, density estimation, conditional hazard rate estimation, classification and censored likelihood models. Figure 14.6 is an example of Locfit local smoothing with a cubic polynomial compared to a binned histogram. Locfit enables computation of both local and global confidence bands. In this case, the figure shows 95% local confidence bands around the smoothed rate estimate.

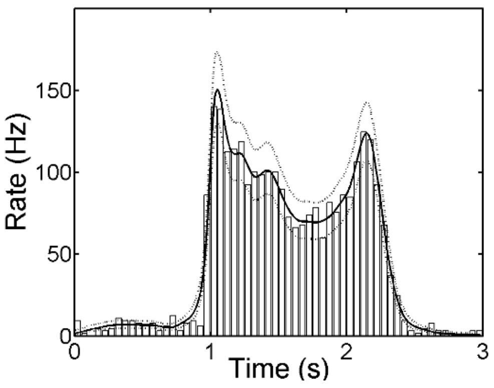


Fig. 14.6. A traditional binned histogram and a Locfit smoothed estimate of the same data set. The Locfit estimate shown uses a nearest-neighbor fraction of 35% when calculating the distribution.

14.7 Repeating synaptic inputs on a single neuron: seeking and finding with Matlab[11]

A single neuron in the mammalian cortex receives synaptic inputs from up to thousands of other neurons. Whole-cell patch-clamp electrophysiological recordings of single neurons reveal many of these synaptic events to the investigator, allowing analyses of the large and active neuronal network that impinges on this single neuron. The goals of the analyses are twofold: to find patterns in this barrage of synaptic inputs, and then determine whether these patterns are produced by design or chance.

While we do not know a priori the patterns to search for, we can still attempt to find patterns. One approach is to search for repeats-that is, sequences of synaptic inputs that repeat later in the recording with significant precision (Figures 14.7, 14.8, 14.9). This is analogous to a study of language by a completely naive observer: a language has a finite vocabulary and set of phrases that can be identified through a search for repeats. This search involves comparing all segments of the recording with itself in an iterative process.

The cross-correlation function is at the heart of this analysis. This function quantifies the temporal similarities of those waveforms and can initially identify whether or not two segments of a long recording might be significantly similar. This initial identification of a repeat is tentative, but the interval at which the repeat is found is stored and examined more carefully in subsequent analyses. Segments that do not pass a minimum threshold are passed over and not analyzed further, saving some time in the subsequent intensive analysis.

[11] Contribution by G. Aaron.

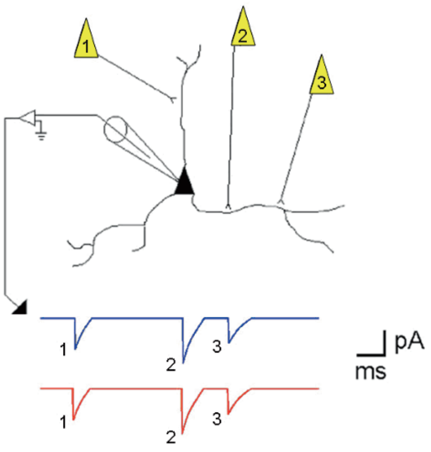
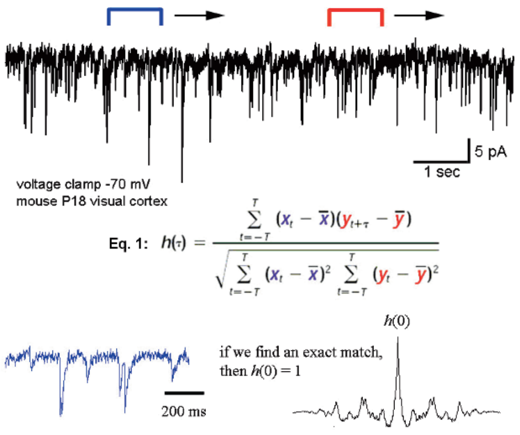
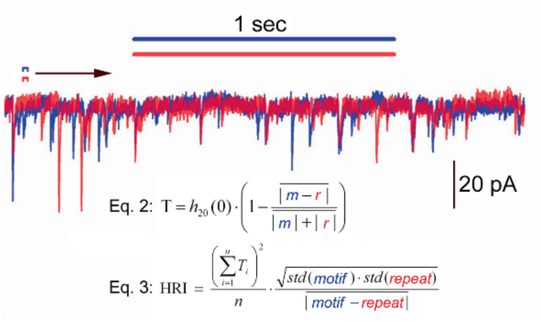


Fig. 14.7. Cartoon of a single neuron recorded within a 4 neuron network. Imagine that the 3 neurons connected to this one recorded neuron fire a sequence of action potentials. This sequence may be reflected in the synaptic currents recorded intracellularly (blue trace). If this same sequence is repeated some time later, the intracellular recording may reflect this repeat of synaptic activity (red trace).



Eq. 1: $$h(\tau) = \frac{\sum_{t=-T}^{T} (x_t - \bar{x})(y_{t+\tau} - \bar{y})}{\sqrt{\sum_{t=-T}^{T} (x_t - \bar{x})^2 \sum_{t=-T}^{T} (y_t - \bar{y})^2}}$$

Fig. 14.8. Searching for repeats. A continuous 10 second stretch intracellularly recording postsynaptic currents (PSCs) is displayed. The program scans this recording, comparing every one second interval with every other one second interval. Here, the blue and red brackets represent these 1 second scanning windows. These one second segments are compared against each other via a cross-correlation equation (Eq. 1). If there were a perfect repeat of intracellular activity, then the correlation coefficient at the zeroth lag time would be 1.0.



Eq. 2: $$T = h_{20}(0) \cdot \left(1 - \frac{\overline{|m-r|}}{\overline{|m|+|r|}}\right)$$

Eq. 3: $$\mathrm{HRI} = \frac{\left(\sum_{i=1}^{n} T_i\right)^2}{n} \cdot \frac{\sqrt{std(motif) \cdot std(repeat)}}{\overline{|motif - repeat|}}$$

Fig. 14.9. Motif-repeat segments that yield a minimum threshold $h(0)$ value are remembered and subsequently analyzed via a high resolution index (HRI). The 1 second segments are aligned according to the best h(0) value, and then they are scanned with a 20 msec time window that computes many cross-correlation values. These correlation values are then adjusted according to the amplitude differences between these short segments (Eq. 2), and values passing a threshold are recorded and used in Eq. 3. The idea of this process is to find very similar PSCs recurring in a precise sequence. Finally, the number of precisely occurring PSCs in a long segment and the precision of those repeats are calculated in the HRI equation (Eq. 3), yielding an index of repeatability.

We used these search programs in analyzing long voltage-clamp intracellular recordings (8 minutes) from slices of mouse visual cortex. These were spontaneous recordings, meaning no stimulation was applied to the slices. Thus, the currents identified in the recordings were presumably the result of synaptic activity, created in large part by the action potential activity of synaptically-coupled neurons. The search algorithms described here were able to find instances of surprising repeatability, as judged by eye (Figure 14.10).

What occurs after the initial identification of putative repeats depends on the hypothesis being tested as well as the recording conditions. If the search is for repeating sequences of postsynaptic currents from a voltage-clamp recording, then the average time course of a single postsynaptic current becomes a critical parameter. Many cross-correlation analyses are performed at these smaller time windows, increasing the sensitivity of the measurement.

Given a long enough recording (several minutes) and many synaptic events, it should be expected that some repeating patterns emerge. The question is then whether the patterns we find are beyond a level that could be expected to occur by chance. In fact, it is undetermined as to whether the specific examples shown in Figure 14.10 are deterministically or randomly generated, although there is strong evidence that non-random patterns do emerge in these recordings ([53], however, see also [72]). The development of surrogate recordings-artificial data that is based on the real-is one way to deal with this issue. These surrogate data can then be compared to the real to see which produces more putative repeats (repeats being judged by the index methods

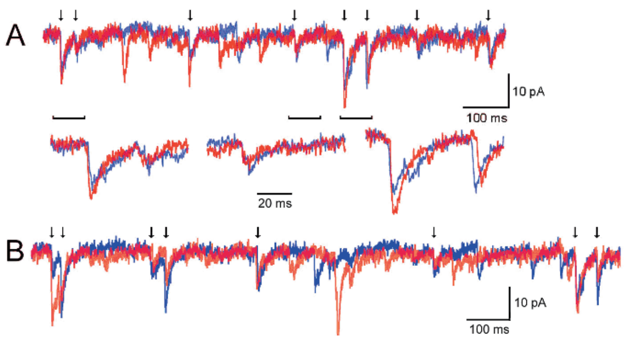


Fig. 14.10. Apparent repeats of synaptic events found. **A:** Two segments from a voltage-clamp recording are displayed, superimposed on each other. The blue trace occurred some time before the red trace, and yet the sequence of synaptic events appear similar. Arrows indicate time points where these synaptic events appear to repeat, and the brackets indicate segments that are temporally expanded below. **B:** Another example of a repeat.

in Figures 14.7 through 14.9). If the real data produces more putative repeats than a large number of generated surrogate data sets, then some evidence is given for a non-random generation of these repeats. The study of this issue is ongoing and involves more than the scope of this chapter. In short, attention must be given to the specific hypothesis tested, as well as the limits and sensitivity of the detector itself (described in Figs 14.7 through 14.9). Perhaps a more convincing method is to perturb the cortex in a biologically relevant manner and see how such manipulations affect or even produce the repeatable patterns (as shown in [63]).

14.8 A method for discovering spatio-temporal spike patterns in multi-unit recordings[12]

14.8.1 Introduction

In previous work we have shown that neurons *in vitro* responded to the repeated injections of somatic current by reliably emitting a few temporal spike patterns differing from each other by only a few spikes [29]. The techniques used to find these patterns were applied to data obtained in vivo from the Lateral Geniculate nucleus in the anesthetized cat [82] and from area MT in the behaving monkey [15]. In these two preparations, the animals were presented with repeated occurrence of the same visual stimulus. Single neurons were recorded and spiking was found to be very reliable when time-locked to the stimulus. In both datasets, however, groups of temporally precise firing

[12] Contribution by J.-M. Fellous.

patterns were identified above and beyond those that were reported. These patterns could last a few hundreds of milliseconds (monkey) to several seconds (cat), and were the result of the complex but largely deterministic interactions between stimulus features, network and intrinsic properties [29]. In principle, this clustering technique can be used on any ensemble of spike trains that have a common time scale: either time locked to the same event (repeated stimulus presentations for one recorded neuron, as above), or recorded simultaneously as in multi-unit recordings. In the first case, groups of trials are determined that share common temporal structures relative to stimulus presentation. In the second case, groups of neurons (called here neural assemblies) are identified that show some temporal correlations.

Finding neural assemblies can be thought of a dimensionality reduction problem: out of N neurons recorded (N-dimensions) how many 'work together'? Standard methods for dimensionality reduction such as principal or independent component analysis have been used on populations of neurons [19, 57]. Such dimensionality reduction however does not typically result in the selection of a subset of neurons, but on a linear combination of their activity (population vector). The weighting of these neurons may or may not be biologically interpretable. Another approach is that of peer prediction, where the influence of $N-1$ recorded neurons onto a single neuron is captured by $N-1$ weights that are 'learned' [44]. The focus of this approach, like that of many information theoretical approaches [74, 84], is to explain a posteriori the firing pattern of neurons, rather than to study their spatio-temporal dynamics. Another approach is that of template matching where the rastergram of multi neuronal activity is binned to generate a sequence of population vectors (a N-dimensional vector per time bin) [23, 73]. A group of T consecutive population vectors is then chosen (NxT matrix, the 'template') and is systematically matched to the rest of the recording. Other methods use the natural oscillations present in the EEG (theta cycle or sharp waves) to define the length of the template [54]. These methods however require the choice of a bin size and the choice of the number of consecutive population vectors (T, another form of binning). We propose here a general method for the detection of transient neural assemblies based on a binless similarity measure between spike trains.

14.8.2 Methods

To illustrate the effectiveness of the technique, we built a biophysical simulation of a small network of neurons using the NEURON simulator [46]. Neurons were single compartments containing a generic leak current, sodium and potassium currents responsible for action potential generation [37], a generic high voltage calcium current [83], a first order calcium pump [25] and a calcium-activated potassium current to control burst firing [25]. In addition, two generic background synaptic noise currents (excitatory and inhibitory) were added to recreate *in vivo* conditions [26, 28, 75]. Synaptic connections were formed with AMPA synapses.

14.8.3 Results

Figure 14.11A shows the typical recording and modeling configuration considered here. Extracellular recording electrodes are lowered near a group of cells. Neurons in this area are interconnected (thin lines, only a few represented for clarity), but some of them form sparse groups linked by stronger synaptic connections (thick black lines). Two of these putative assemblies are depicted in gray and black. The extracellular recording electrodes have typically access to only a fraction of these assemblies (ellipse, 3 neurons each). To illustrate this configuration further, we built a biophysical simulation of fifteen principal neurons containing 2 different assemblies of 5 neurons each. Figures 14.11B and 14.11C show representative traces of the membrane voltage of two neurons: One that does not belong to an assembly and one that does (top and bottom respectively). No difference can be detected in the statistics of the membrane potential between these neurons (average, standard deviation, coefficient of variation of spiking and firing rate were tested).

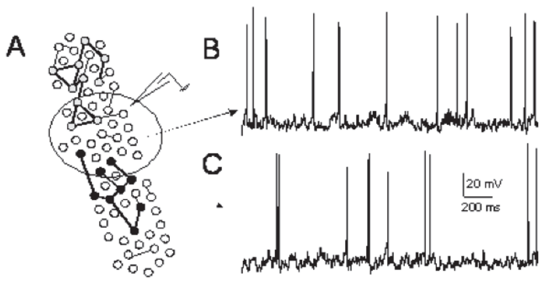


Fig. 14.11. Simulations of neural assemblies. **A:** Schematic representation of a recording configuration in CA3. Panels **B** and **C** show the simulated membrane potential of a cell outside an assembly (**B**) and of a cell within one of the two assembles (**C**, black neuron).

Figure 14.12A shows a rastergram of 2 seconds of spontaneous activity of all 15 neurons. The simulation is tuned so that all neurons have the same mean firing rate (7.5 Hz). Again, no difference can be seen between the spike trains of neurons belonging to an assembly (filled circles labeled on the left of the Y-axis), and those that do not (white circles). Such rastergrams are typically the only neurophysiological information available from multi-unit recordings in the behaving animal. Experimentally, the labels are of course unknown and the goal of this work is to identify the 'hidden' assemblies on the basis of unlabeled rastergrams.

To do so, we first build a similarity matrix computed on the basis of a binless correlation measure used previously [93] (Figure 14.12B). Each square i,j of this 15x15 matrix represents the similarity between the spike trains of neurons i and j (similarities vary between 0 (dissimilar-black) and 1 (identical-white)). A fuzzy clustering method is then applied to this matrix and rows and columns are re-ordered under the constraint that the matrix remains symmetric [29] (Figure 14.12C). Groups of highly similar neurons are now apparent (white regions in the upper left and lower right corners of the matrix). The algorithm identified 3 areas in this matrix (dashed lines). Since each row represents the similarities of one neuron to the 14 others, the matrix reordering can be applied to the spike trains themselves, and the re-ordered rastergram is shown in panel D. The assemblies are correctly identified as evidenced by the grouping of the symbols on the left (assemblies: filled circle, other neurons: white circles). Cluster strengths were 2.4 (black circles), 2.2 (gray circles) and 1.2 (white circles), making the first two clusters significant (above 1.5 [29]). In this simulation, and for illustrative purposes, the two assemblies were implemented in a different manner. The first assembly (gray neurons in Figures 14.11 and 14.12) contained neurons that were correlated because of slightly stronger AMPA intrinsic connections between them. This assembly formed transient assembly-wide correlations (ellipses in Figure 14.12D). The second assembly (black neurons in Figures 14.11 and 14.12) contained neurons that were correlated because they received a common modulation of their mean excitatory background inputs. Unlike with the first assembly, this assembly did not form assembly-wide correlations.

The algorithm successfully identified the assemblies in both cases, showing that it was not sensitive to the source of the correlations and did not require all the neurons within an assembly to be simultaneously correlated. The performance of the clustering algorithm in detecting functionally connected cells, when the connection strength is systematically varied was assessed. The network is analogous to that of Figure 14.11 with gray neurons only. Our simulations (50 independent simulations per connection strength) show that the detection is significantly above chance for connection values that yield probability of postsynaptic firing due to a single presynaptic spike as low as 10%.

The performance of the clustering algorithm was also assessed in the case where the neurons in Figure 14.11A are interconnected by weak synapses (2%) but receive part of their average background inputs from a common source (i.e. black neurons only). The common input was simulated as a modulation of X% of the mean background excitatory synaptic noise, where X is systematically varied. The time course and frequency content of the modulation is obtained from white-noise filtered by AMPA-like alpha-function [65]. Note that in these simulations, the overall firing rate of all cells in the assembly were kept constant and identical to that of the cells outside the assembly, so as to not bias the algorithm into clustering cells by firing rates rather than by temporal correlations. Previous work has however shown that a properly

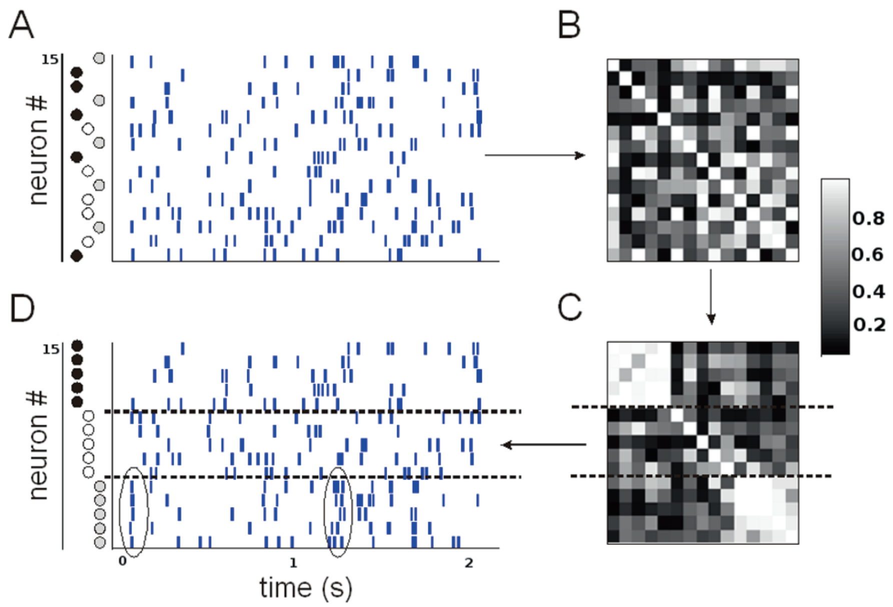


Fig. 14.12. Detection of cell assemblies using Spike Train Clustering. **A**: Original rastergram obtained from a biophysical simulation of 15 neurons. Two assemblies of five neurons are 'hidden' (filled circles). **B**: Similarity matrix based on the whole rastergram. **C**: Reordering of the similarity matrix after fuzzy clustering with 3 clusters [29]. Two significant clusters are apparent in the upper left and lower right corners. Similarity gray scale applies to B and C. **D**: The same reordering in C is applied to the rastergram. The hidden assemblies are completely recovered (grouping of labels).

tuned similarity measure can make the clustering algorithm relatively insensitive to heterogeneous baseline firing rates [29]. Simulations (50 independent simulations per X) show that a modulation of 15% of the background inputs can be effectively detected above chance.

14.8.4 Conclusions

Various methods for multiple spike train analysis have been reviewed elsewhere [12, 14]. In general, the detection of neural assemblies can be accomplished in two steps: 1) Computation of the similarity between the spike trains of the recorded neurons, 2) Use of these similarities to detect eventual groups of neurons that are more similar to each other than they are to the other neurons.

The traditional measures of similarity between spike trains include the cross-correlation coefficient, its generalization with the Joint Peristimulus Time Histograms (where Pearson correlations between two neurons are computed at different time lags) [2, 107] and the cross-intensity function (where estimation of the firing rate of one neuron is made relative to that of another at different time lags) [12]. These methods require some form of binning which involves the a priori choice of a time scale of integration (bin size) and the choice of bin boundaries (with or without overlaps), both of which may introduce artifacts. There is some evidence that computing correlation in the frequency domain (i.e. cross coherence) may help reduce the sensitivity to bin size [76, 99].

Two other concerns arise 1) Correlation coefficients may introduce artificial correlations due to bursting, which is common in many brain areas and 2) correlations are computed on the basis of pairs of neurons, and are unable to capture spike patterns which consist in P neurons firing in one of several possible orders (i.e. in an assembly of P neurons, one neuron may fire before a second at one point in time, or after that neuron later on depending on the firing of the other P-2 neurons) [29]. Because the order of firing between two neurons may not be always the same, the pair wise correlation coefficient could become insignificant.

A few methods have attempted to go beyond pairwise assessments by accounting for correlations between 3 or 4 neurons [1, 24, 52], but those methods are not easily generalizable to 50-100 neurons, which is the typical number of neurons recorded simultaneously. Other methods such as Bayesian estimations [4, 13, 68, 106] or unitary event analysis (statistical detection of spike coincidences that occur above chance) [42, 43] are statistical in nature, and are not suitable for the dynamic assessment of patterns in time. The work we presented here however has that potential [59]. In sum, new methods for detecting spike patterns (i.e. high order correlations) such as that proposed here and ways of comparing their similarities are sorely needed.

14.9 Summary

The authors of this chapter hope that these mini-proceedings will be useful to all those who are already involved in the field of electrophysiology and are looking for currently available data analysis tools and techniques. However, we also hope that Computational Intelligence researchers will find it valuable as their interest will be sparked by the presented discussion of the challenges prevalent in electrophysiology.

The organizers of the workshop would like to thank all the contributors and participants for their invaluable input and involvement in the creation of these mini-proceedings.

References

1. Abeles M, Gat I (2001) Detecting precise firing sequences in experimental data. J Neurosci Methods 107:141–54

2. Aertsen AM, Gerstein GL, Habib MK, Palm G (1989) Dynamics of neuronal firing correlation: modulation of “effective connectivity”. J Neurophysiol 61:900–17
3. Baker S, Baseler H, Klein S, Carney T (2006) Localizing sites of activation in primary visual cortex using visual-evoked potentials and functional magnetic resonance imaging. J Clinical Neurophys. 23(5):404–15
4. Barbieri R, Frank LM, Nguyen DP, Quirk MC, Solo V, Wilson MA, Brown E (2004) A Bayesian decoding algorithm for analysis of information encoding in neural ensembles. Conf Proc IEEE Eng Med Biol Soc 6:4483–86
5. Berners-Lee T, Hall W, Hendler J, Shadbolt N, Weitzner DJ (2006) Creating a Science of the Web. Science 313(5788):769–71
6. Bhalla US and Bower JM (1993) Exploring parameter space in detailed single neuron models: Simulations of the mitral and granule cells of the olfactory bulb. J Neurophysiol 69:1948–65
7. Bloom F, et al. (2003) Neuroscience Database Gateway.
Available: `http://ndg.sfn.org`
8. Bokil H, Pesaran B, Andersen RA, Mitra PP (2006) A method for detection and classification of events in neural activity. IEEE Transactions on Biomedical Engineering 53:1678–87
9. Bokil H, Purpura K, Schofflen J-M, Thompson D, Pesaran B, Mitra PP (2006) Comparing spectra and coherences for groups of unequal size. J Neurosci Methods 159:337–45
10. Bokil H, Tchernichovski O, Mitra PP (2006) Dynamic phenotypes: Time series analysis techniques for characterising neuronal and behavioral dynamics. Neuroinformatics Special Issue on Genotype-Phenotype Imaging in Neuroscience 4:119–28
11. Bower JM and Beeman D (1997) The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System. Springer
12. Brillinger DR (1992) Nerve cell spike train data analysis: A progression of technique. Journal of the American Statistical Association 87:260–71
13. Brown EN, Frank LM, Tang D, Quirk MC, Wilson MA (1998) A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. J Neurosci 18:7411–25
14. Brown EN, Kass RE, Mitra PP (2004) Multiple neural spike train data analysis: state-of-the-art and future challenges. Nat Neurosci 7:456–61
15. Buracas GT, Zador AM, DeWeese MR, Albright TD (1998) Efficient discrimination of temporal patterns by motion-sensitive neurons in primate visual cortex. Neuron 20:959–69
16. Cannon RC and D’Alessandro G (2006) The ion channel inverse problem: neuroinformatics meets biophysics. PLoS Comput Biol 2(8):e91
17. Carmena JM, Lebedev MA, Henriquez CS, Nicolelis MAL (2005) Stable ensemble performance with single-neuron variability during reaching movements in primates. J Neurosci 25(46):10712–16
18. Chance FS, Nelson SB, Abbott LF (1998) Synaptic Depression and the Temporal Response Characteristics of V1 Cells. J Neurosci 18:4785–99
19. Chapin JK, Nicolelis MA (1999) Principal component analysis of neuronal ensemble activity reveals multidimensional somatosensory representations. J Neurosci Methods 94:121–40

20. Clancy CE and Rudy Y (2002) Na(+) channel mutation that causes both Brugada and long-QT syndrome phenotypes: a simulation study of mechanism. Circulation 105:1208–13
21. Clancy CE and Kass RS (2004) Theoretical investigation of the neuronal Na+ channel SCN1A: abnormal gating and epilepsy. Biophys J 86:2606–14
22. Crook S, Gleeson P, Howell F, Svitak J, Silver RA (2007) MorphML: Level 1 of the NeuroML standards for neuronal morphology data and model specification. Neuroinf. 5(2):96–104
23. Crowe DA, Averbeck BB, Chafee MV, Georgopoulos AP (2005) Dynamics of parietal neural activity during spatial cognitive processing. Neuron 47:885–91
24. Czanner G, Grun S, Iyengar S (2005) Theory of the snowflake plot and its relations to higher-order analysis methods. Neural Comput 17:1456–79
25. Destexhe A, Contreras D, Sejnowski TJ, Steriade M (1994) A model of spindle rhythmicity in the isolated thalamic reticular nucleus. J Neurophysiol 72:803–18
26. Destexhe A, Rudolph M, Fellous JM, Sejnowski TJ (2001) Fluctuating synaptic conductances recreate in vivo-like activity in neocortical neurons. Neuroscience 107:13–24
27. Fee MS, Mitra PP, Kleinfeld D (1996) Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-Gaussian variability. J Neurosci Methods 69:175–88
28. Fellous J-M, Rudolph M, Destexhe A, Sejnowski TJ (2003) Variance detection and gain modulation in an in-vitro model of in-vivo activity. Neuroscience 122:811–29
29. Fellous JM, Tiesinga PH, Thomas PJ, Sejnowski TJ (2004) Discovering spike patterns in neuronal responses. J Neurosci 24:2989–3001
30. Gardner D, Abato M, Knuth KH, DeBellis R, Erde SM (2001) Dynamic publication model for neurophysiology databases. Philos Trans R Soc Lond B Biol Sci 356(1412):1229–47
31. Gardner D, Knuth KH, Abato M, Erde SM, White T, DeBellis R, Gardner EP (2001) Common data model for neuroscience data and data model exchange. J Am Med Inform Assoc 8:17–33
32. Gardner D (2004) Neurodatabase.org: networking the microelectrode, Nat Neurosci 7(5):486–87
33. Gardner D (2004) *personal communication*
34. Gleeson P, Steuber V, Silver RA (2007) neuroConstruct: a tool for modeling networks of neurons in 3D space. Neuron 54(2):219–35
35. Goddard NH, Hucka M, Howell F, Cornelis H, Shankar K, Beeman D (2001) Towards NeuroML: model description methods for collaborative modelling in neuroscience. Philos Trans R Soc Lond B Biol Sci 356(1412):1209-1228
36. Goddard NH, Cannon RC, Howell FW (2003) Axiope tools for data management and data sharing. Neuroinformatics 1(3):271–84
37. Golomb D, Amitai Y (1997) Propagating neuronal discharges in neocortical slices: computational and experimental study. J Neurophysiol 78:1199–211
38. Golowasch J, Abbott LF, Marder E (1999) Activity-dependent regulation of potassium currents in an identified neuron of the stomatogastric ganglion of the crab Cancer borealis. J Neurosci 19(20):RC33
39. Golowasch J, Goldman MS, Abbott LF, Marder E (2002) Failure of averaging in the construction of a conductance-based neuron model. J Neurophysiol 87(2):1129–31

40. Gonzalez-Heydrich J, Steingard RJ, Putnam F, Beardslee W, Kohane IS (1999) Using 'off the shelf' computer programs to mine additional insights from published data: diurnal variation in potency of ACTH stimulation of cortisol secretion revealed. Comput Methods Programs Biomed 58(3):227–38
41. Greenstein JL, Wu R, Po S, Tomaselli GF, Winslow RL (2000) Role of the calcium-independent transient outward current I(to1) in shaping action potential morphology and duration. Circ Res 87:1026-1033
42. Grun S, Diesmann M, Aertsen A (2002) Unitary events in multiple single-neuron spiking activity: I. Detection and significance. Neural Comput 14:43–80
43. Grun S, Diesmann M, Aertsen A (2002) Unitary events in multiple single-neuron spiking activity: II. Nonstationary data. Neural Comput 14:81–119
44. Harris KD, Csicsvari J, Hirase H, Dragoi G, Buzsaki G (2003) Organization of cell assemblies in the hippocampus. Nature 424:552–6
45. Herzog RI, Cummins TR, Waxman SG (2001) Persistent TTX-resistant Na+ current affects resting potential and response to depolarization in simulated spinal sensory neurons. J Neurophysiol 86(3):1351–64
46. Hines ML and Carnevale NT (1997) The NEURON simulation environment. Neural Comput 9(6):1179–209
47. Hines ML, Morse T, Migliore M, Carnevale NT, Shepherd GM (2004) ModelDB: A database to support computational neuroscience. J Comput Neurosci 17(1):7–11
48. Hodgkin AL and Huxley AF (1952) A quantitative description of membrane current and its application to conduction and excitation in nerve. J Physiol Lond 117:500-544
49. Hulata E, Segev R, Ben-Jacob E (2002) A method for spike sorting and detection based on wavelet packets and Shannon's mutual information. J Neurosci Methods 117(1):1–12
50. Humphries MD, Stewart RD, Gurney KN (2006) A physiologically plausible model of action selection and oscillatory activity in the basal ganglia. J Neurosci 26(50):12921–42
51. Huyser K and van der Laan J (1992) Data Thief v.1.0.8. `Available:http://www.datathief.org/`
52. Iglesias J, Villa AE (2007) Effect of stimulus-driven pruning on the detection of spatiotemporal patterns of activity in large neural networks. Biosystems 89:287–93
53. Ikegaya Y, Aaron G, Cossart R, Aronov D, Lampl I, Ferster D, Yuste R (2004) Synfire chains and cortical songs: temporal modules of cortical activity. Science 304(5670):559–64
54. Jackson JC, Johnson A, Redish AD (2006) Hippocampal sharp waves and reactivation during awake states depend on repeated sequential experience. J Neurosci 26:12415–26
55. Kennedy DN and Haselgrove C (2006) The internet analysis tools registry: a public resource for image analysis. Neuroinformatics 4(3):263–70
56. Kuzmick V, Lafferty J, Serfass A, Szperka D, Zale B, Nagvajara P, Johnson J, Moxon K (2001) Novel epileptic seizure detection system using multiple single neuron recordings. Conf Proc IEEE 27th Ann Northeast Bioeng, pp. 7–8
57. Laubach M, Shuler M, Nicolelis MA (1999) Independent component analyses for quantifying neuronal ensemble interactions. J Neurosci Methods 94:141–54

58. Leblois A, Boraud T, Meissner W, Bergman H, Hansel D (2006) Competition between feedback loops underlies normal and pathological dynamics in the basal ganglia. J Neurosci 26(13):3567–83
59. Lipa P, Tatsuno M, Amari S, McNaughton BL, Fellous JM (2006) A novel analysis framework for characterizing ensemble spike patterns using spike train clustering and information geometry. Society for Neuroscience Annual Meeting. Atlanta, GA, pp. 371–76
60. Llinas RR, Ribary U, Jeanmonod D, Kronberg E, Mitra PP (1999) Thalamocortical dysrhythmia: A neurological and neuropsychiatric syndrome characterized by magnetoencephalography. Proceedings of the National Academy Of Sciences of The United States of America 96:15222–27
61. Loader C (1999) Local Regression and Likelihood. Springer
62. Lytton WW (2006) Neural query system - data-mining from within the neuron simulator. Neuroinformatics 4(2):163–75
63. MacLean JN, Watson BO, Aaron GB, Yuste R (2005) Internal dynamics determine the cortical response to thalamic stimulation. Neuron 48:811–23
64. Magee JC (1998) Dendritic hyperpolarization-activated currents modify the integrative properties of hippocampal CA1 pyramidal neurons. J Neurosci 18(19):7613-7624
65. Mainen ZF, Sejnowski TJ (1995) Reliability of spike timing in neocortical neurons. Science 268:1503–6
66. Mamlouk AM, Sharp H, Menne KML, Hofmann UG, Martinetz T (2005) Unsupervised spike sorting with ICA and its evaluation using GENESIS simulations. Neurocomputing, 65-66:275–82
67. Markram H (2006) The blue brain project. Nat Rev Neurosci 7(2):153-160
68. Martignon L, Deco G, Laskey K, Diamond M, Freiwald W, Vaadia E (2000) Neural coding: higher-order temporal patterns in the neurostatistics of cell assemblies. Neural Comput 12:2621–53
69. Migliore M and Shepherd GM (2002) Emerging rules for the distributions of active dendritic conductances. Nat Rev Neurosci 3(5):362–70
70. Migliore M and Shepherd GM (2005) Opinion: an integrated approach to classifying neuronal phenotypes. Nat Rev Neurosci 6(10):810–18
71. Mitra PP and Pesaran B (1999) Analysis of dynamic brain imaging data. Biophysical J 76:691–708
72. Mokeichev A, Okun M, Barak O, Katz Y, Ben-Shahar O, Lampl I (2007) Stochastic emergence of repeating cortical motifs in spontaneous membrane potential fluctuations in vivo. Neuron 53:413–25
73. Nadasdy Z, Hirase H, Czurko A, Csicsvari J, Buzsaki G (1999) Replay and time compression of recurring spike sequences in the hippocampus. J Neurosci 19:9497–507
74. Nirenberg S, Carcieri SM, Jacobs AL, Latham PE (2001) Retinal ganglion cells act largely as independent encoders. Nature 411:698–701
75. Paré D, Shink E, Gaudreau H, Destexhe A, Lang EJ (1998) Impact of spontaneous synaptic activity on the resting properties of cat neocortical pyramidal neurons In vivo. J Neurophysiol 79:1450–60
76. Percival DB, Walden AT (2000) Wavelet Methods for Time Series Analysis. Cambridge University Press

77. Pesaran B, Pezaris JS, Sahani M, Mitra PP, Andersen RA (2002) Temporal structure in neuronal activity during working memory in macaque parietal cortex. Nature Neuroscience 5:805–11
78. Pittendrigh S and Jacobs G (2003) NeuroSys: a semistructured laboratory database. Neuroinformatics 1(2):167–76
79. Prinz AA, Billimoria CP, Marder E (2003) Alternative to hand-tuning conductance-based models: construction and analysis of databases of model neurons. J Neurophysiol 90:3998–4015
80. Qi W and Crook S (2004) Tools for neuroinformatic data exchange: An XML application for neuronal morphology data. Neurocomp 58-60C:1091–5
81. Rall W (1977) Core Conductor theory and cable properties of neurons. Handbook of Physiology, Sec. 1, The Nervous System, vol. 1, Bethesda, MD: Am Physiol Soc, pp. 39–97
82. Reinagel P, Reid RC (2002) Precise firing events are conserved across neurons. J Neurosci 22:6837–41
83. Reuveni I, Friedman A, Amitai Y, Gutnick MJ (1993) Stepwise repolarization from Ca2+ plateaus in neocortical pyramidal cells: evidence for nonhomogeneous distribution of HVA Ca2+ channels in dendrites. J Neurosci 13:4609–21
84. Rieke F, Warland D, de Ruyter van Steveninck R, Bialek W (1997) Spikes: Exploring the Neural Code. The MIT Press
85. Roth A and Hausser M (2001) Compartmental models of rat cerebellar Purkinje cells based on simultaneous somatic and dendritic patch-clamp recordings. J Physiol 535:445–72
86. Rudy Y, Silva JR (2006) Computational biology in the study of cardiac ion channels and cell electrophysiology. Quart Rev Biophysics 39(1):57–116
87. Rutishauser U, Schuman EM, Mamelak AN (2006) Online detection and sorting of extracellularly recorded action potentials in human medial temporal lobe recordings, in vivo. J Neurosci Methods 154(1-2):204–24
88. Sakmann B and Neher E (1995) Single-Channel Recording (2^{nd} ed). Plenum Press
89. Santhakumar V, Aradi I, Soltesz I (2005) Role of mossy fiber sprouting and mossy cell loss in hyperexcitability: a network model of the dentate gyrus incorporating cell types and axonal topography. J Neurophysiol 93:437–53
90. Santhanam G, Sahani M, Ryu S, Shenoy K (2004) An extensible infrastructure for fully automated spike sorting during online experiments. Conf Proc IEEE Eng Med Biol Soc, pp. 4380–4
91. Schaefer AT, Helmstaedter M, Sakmann B, Korngreen A (2003) Correction of conductance measurements in non-space-clamped structures: 1. Voltage-gated k(+) channels. Biophys J 84:3508–28
92. Schaefer AT, Helmstaedter M, Schmitt AC, Bar-Yehuda D, Almog M, Ben-Porat H, Sakmann B, et al. (2007) Dendritic voltage-gated K+ conductance gradient in pyramidal neurones of neocortical layer 5B from rats. J Physiol 579:737–52
93. Schreiber S, Fellous J-M, Tiesinga PH, Sejnowski TJ (2003) A new correlation-based measure of spike timing reliability. Neurocomputing 52-54:925–31
94. Surkis A, Peskin CS, Tranchina D, Leonard CS (1998) Recovery of cable properties through active and passive modeling of subthreshold membrane responses from laterodorsal tegmental neurons. J Neurophysiol 80(5):2593–607

95. Suzuki N, Takahata M, Sato K (2002) Oscillatory current responses of olfactory receptor neurons to odorants and computer simulation based on a cyclic AMP transduction model. Chem Senses 27:789–801
96. Taylor AL, Hickey TJ, Prinz AA, Marder E (2006) Structure and visualization of high-dimensional conductance spaces. J Neurophysiol 96:891–905
97. Terman D, Rubin JE, Yew AC, Wilson CJ (2002) Activity patterns in a model for the subthalamopallidal network of the basal ganglia. J Neurosci 22(7):2963–76
98. Tchernichovski O, Nottebohm F, Ho CE, Pesaran B, Mitra PP (2000) A procedure for an automated measurement of song similarity. Animal Behaviour 59:1167–76
99. Thomson DJ, Chave AD (1991) Jacknifed error estimates for spectra, coherences, and transfer functions. In: Advances in spectrum analysis and array processing (Haykin S, ed), pp. 58-113. Prentice Hall, Englewood Cliffs, NJ
100. Tóth TI and Crunelli V (2001) Estimation of the activation and kinetic properties of INa and IK from the time course of the action potential. J Neurosci Meth 111(2):111–26
101. Traub RD, Contreras D, Cunningham MO, Murray H, Lebeau FE, Roopun A, Bibbig A, et al. (2005) A single-column thalamocortical network model exhibiting gamma oscillations, sleep spindles and epileptogenic bursts. J Neurophysiol 93(4):2194–232
102. Vanier MC and Bower JM (1999) A comparative survey of automated parameter-search methods for compartmental neural models. J Comput Neurosci 7(2):149–71
103. Venkataramanan L and Sigworth FJ (2002) Applying Hidden Markov Models to the Analysis of Single Ion Channel Activity. Biophys J 82:1930–42
104. Weaver CM and Wearne SL (2006) The role of action potential shape and parameter constraints in optimization of compartment models. Neurocomputing 69:1053–57
105. Womelsdorf T, Fries P, Mitra PP, Desimone R (2006) Gamma-band synchronization in visual cortex predicts speed of change detection. Nature 439:733–36
106. Zhang K, Ginzburg I, McNaughton BL, Sejnowski TJ (1998) Interpreting neuronal population activity by reconstruction: unified framework with application to hippocampal place cells. J Neurophysiol 79:1017–44
107. Zohary E, Shadlen MN, Newsome WT (1994) Correlated neuronal discharge rate and its implications for psychophysical performance. Nature 370:140–43