September 6, 2004

XNBC *v*8

A Workstation for

Biological Neural Network Simulation

Reference manual updated up to versions 8.3x

Jean-François VIBERT
B3E ESI ISARS – INSERM U444
CHU Saint-Antoine,
Université Pierre et Marie Curie (Paris VI)
Paris
FRANCE
vibert@b3e.jussieu.fr

Fabián ALVAREZ
Facultad de Ciencias
Sección Neurociencias
Montevideo
URUGUAY
fapa@fcien.edu.uy

The XNBC project was supported by the DRET (Contract 91/1246A and 94/2526A).

Contents

List of Figures

Chapter 1

Presentation of XNBC

XNBC is a simulation workstation for neurobiologists developed for research purpose.

XNBC means X Window Neuro_Bio_Clusters. *X Window System* is the Unix windowing system, Neuro_Bio stands for biological neurons, and Clusters for the way neurons are grouped. XNBC is thus a workstation for biological neural network simulation. This software is a kit of different complementary coherent tools, created by neuroscientists who needed for their research. These tools were designed to simulate biological neural networks, and to analyze their results, in order to test hypotheses.

XNBC provides two neuron editors, intended for two simulation levels: simple and fast neuron models (phenomenological neuron), and sophisticated but slower models (conductance based neuron). The phenomenological neuron editor implements in fact three models: the classical *Leaky Integrator Model* (LIM), an enhanced version of the former, *The Phenomenological Unit Model* (PUM), and the *Burster Unit Model (BUM)*. The three take into account two variables (membrane potential and threshold variables), and present several parameters which may be adjusted. The *Conductance Based Model* (CBM) involves the conductance variations in different ionic channels (14 at present). Furthermore, XNBC provides a *Virtual* neuron mode, which allows to take into account data stored in a file, coming from either a live experiment or a previous simulation.

XNBC allows to build a large variety of neural networks, by using neurons grouped at two levels: nuclei and clusters. A nucleus is a collection of neurons spatially related; a cluster is a group of neurons which share identical membrane properties (which can pertain or not to the same nucleus). The networks may be build, either with a simple editor, without caring about precise spatial coordinates of neurons, or the nuclei and clusters can be positioned according to the Horsley–Clarke coordinate system (in three dimensions). In the latter case, the neuroscientist can create intra and inter nuclei or cluster connections, according to his experimental protocol.

Once the basic neuron (or neurons) is (are) defined, and then grouped into the network, the simulation is ready to begin. The simulator takes charge of the temporal evolution of the totality of the neuron's variables. In addition, perturbations (noise, stimulation, modulation, drugs), can be done either at the level of the nuclei or at the level of the clusters, allowing a high flexibility on the simulated experiment control. XNBC computes and saves the simulation results with different file format options.

Finally, XNBC may display the temporal evolution of the network, or those of selected neurons, and also point process, frequency and dynamic analyzes can be performed.

1.1 The XNBC components

The XNBC simulation toolkit is composed of several independent tools, integrated by the `xnbc` program that presents a control panel from which all the process are controlled. The XNBC simulation toolkit is made up of several programs:

- `xnbc` is the control program which organizes your work and launches each one of the above programs
- two graphic neuron editors to adjust the neuron parameters
  - `P_neuron` is the phenomenological neuron editor, which allows to define neurons based on the three pre-defined models (LIM, PUM, and BUM)
  - `G_neuron` is the conductance based neuron editor, which allows to define CBM neurons
- two graphic network editors
  - `link_edit` builds simple networks, from a single neuron up to a large number of linked nuclei, but each nucleus may be build up only by identical neurons
  - `network_editor` builds complex and anatomically precisely tuned networks (different neurons can be mixed into a nucleus)
- `drug_editor` allows to “design” drugs with effects on specific ionic currents
- `nbc_x` is the simulator itself which iterates the temporal evolution of the totality of the neurons of the network
- `visu_nbc` is a visualization tool to examine in several ways the simulation results
- two analysis tools
  - `xtms` is a time series analysis tool with 31 different analyzes
  - `xcaa` is nucleus activity analysis tool with 8 different analyzes

1.2 The XNBC objects

1.2.1 The neuron

The neuron is the basic element of XNBC. It represents a neural entity that has a particular physical type, and has a specified location. A neuron can share its membrane properties with other neurons (we say that they pertain to the same cluster –see below–). Nevertheless, each neuron is individually simulated and has its own temporal variables evolution (membrane potential, ionic conductance, etc...), hence each one has its own life. Furthermore, neurons can be anatomically positioned in the 3D space if needed (see 1.2.3).

1.2.2 The cluster

The concept of cluster has proven to be a very powerful one, in order to describe large sets of neurons and to group them. When describing the cluster properties, we describe in first term the neuron model (LIM, BUM, CBM or Virtual), and then the basic parameters of the neurons. These parameters include membrane properties (rest potential, rest threshold, membrane capacity, mean Na or K conductance, etc) and input properties (EPSP size, membrane noise).

At present, four different ways of modeling the neurons are available in XNBC, and thus, are the units available to define the clusters:

LIM the Leaky Integrator Model of neuron, the classical simple leaky integrator (cluster properties and its default values in App. C.2

PUM the Phenomenologic Model of neuron, an enhanced leaky integrator with adaptation and post spike membrane shunt (cluster properties and its default values in App. B.2

BUM a phenomenologic model of conditional burster with adaptation and post spike membrane shunt (cluster properties and its default values in App. D.2

CBM a Hodgkin-Huxley (HH) like model with 14 different transmembranar currents (cluster properties and its default values in App. E.2

Virtual not simulated by the simulator, data stored in a file and coming from either a live experiment or a previous simulation. This model allows to build hybrid networks made of simulated neurons receiving inputs from neurons experimentally recorded.

Note that the cluster properties may be identified to the neuron ones, for which it is equivalent to refer them as cluster or neuron properties.

When the network is built, the cluster allows to take into account:

- the connection according to the inter neuron distance (interneuronal delays)
- the connection pattern (synaptic weights array)

1.2.3 The nucleus

The nucleus is a new concept, introduced with XNBC V8.0. This concept introduces the spatial influence in the networks interactions. It allows to consider groups of neurons that have the same location area, specified by a center and a radius around this center. The nucleus allows to take into account:

- the anatomic location of neurons (the Horsley Clarke coordinates can be used)
- the connection according to the inter neuron distance (interneuronal delays)
- the connection pattern (synaptic weights array)
- the dissociation of anatomical location and unit characteristics

- the neuromodulator or drug concentration according to the production or injection locus (in a future version of XNBC)

A nucleus is constituted by a given number of neurons. These neurons can pertain to one or several clusters, and, in turn, clusters may span several nuclei (because they describe the neuron behavior, not the neuron location). Neurons inside nuclei can be connected together and to the other nuclei.

When nuclei contain a single cluster, nucleus and cluster can be viewed as equivalent. In this case, if the spatial coordinates are not considered, the simple network editor should be used.

1.2.4 The network

The network is the object for which the simulator computes its temporal evolution. The network may range from a single neuron to a network made up of some hundred of neurons (or even a few thousand if you have a fast computer and time).

Hence, the output of the network editor is made of one or several nuclei and/or one or several isolated neurons. Nuclei and neurons can be anatomically positioned if necessary. Nuclei and neurons are connected together by links which represent the axons of constituting neurons.

1.2.5 The connections or links

Neurons have to be connected. The connections can be either excitatory, inhibitory or with NMDA (long lasting excitation). The set of the link values between all the neurons is stored in the synaptic weight array. For instance, a synaptic weight (an element of the array) of zero means that these two neurons are not connected.

Analogously, the values for interneural transmission time of action potentials, called interneural delay (also referred to as axon length), are stored as an array.

The network editors allow to manipulate those arrays. They can be defined either globally or individually, neuron to neuron. In addition, random connection arrays may be created (a mix of excitatory and inhibitory links, or variable axonal delays).

1.3 How to use XNBC

Efforts were made to provide an ergonomic and user friendly user interface, in order to allow neuroscientists without any expertise in computer sciences, to use XNBC.

Once the main program `xnbc` is executed, it launches the control panel, which displays pushbuttons arranged to guide the user to perform sequentially neuron and network definition, simulations and analyses. Nevertheless, the very first thing to do is to choose a name for the simulation (menu 'Experiment->New'). This name will be used as a generic base name, for which `xnbc` creates a directory with the name `chosen_name_nbc`, in which all files created during the simulation session are saved. Hence, related files keep together and isolated from those of other simulations.

Then, the user must sequentially:

A choose the parameters of the neuron models,

B build a neural network

C prepare new drug definitions if necessary

D run the simulation

E visualize the simulation results

F analyze the simulated network behavior.

These five steps are sketched below, and detailed descriptions of their respective programs are supplied for each one on separate chapters.

Besides that, the simulator itself (`nbc_x`, described in chapter 8) controls all the simulation process, using pull-down menus instead of graphic pushbuttons. This feature allows to modify neuron and network parameters "on the fly" during the simulation.

1.3.1 Choose the parameters of the neuron models,

Four neuron models are available, the phenomenologic ones (LIM, PUM, and BUM), and the CBM (a HH type model with 14 different ionic currents). Data stored in files from actual experiments allows to make "hybrid" networks, for which these data are considered as a fifth model (Virtual neuron).

The modeled neurons are created using graphic tools. During the simulation, neurons parameters may be modified in order to mimic electrical stimulations and drugs action.

There exist two graphic editors to adjust the neuron parameters.

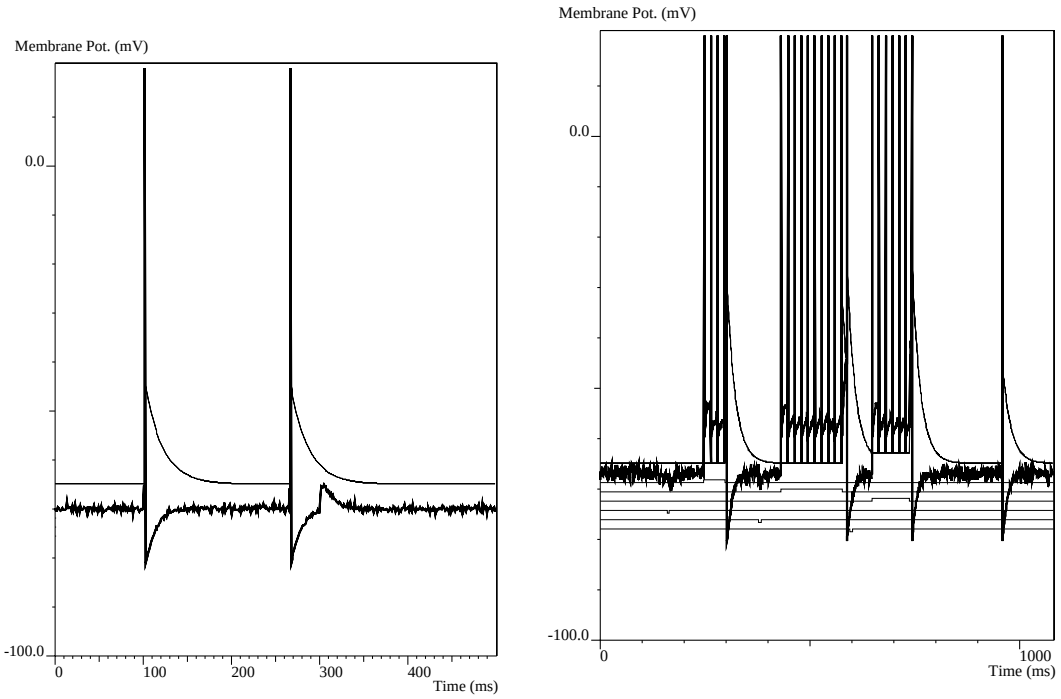
The phenomenological neuron models graphic editor

The editor allows to model the classical a LIM, enhanced with possibility to have threshold adaptation (set at 0 by default). This well-known model had different enhancements along *XNBC* development, giving rise to the PUM. XNBC's PUM implements additional properties, such as fatigue and post-spike membrane shunt (for a complete description of this model, see Vibert *et al.*, 1994).

The user adjusts the parameters by sliding scale cursors, while the temporal evolution of the membrane potential changes in real time according to the parameter values. A simulation as simple as setting the threshold value below the resting potential one, exemplifies how to obtain a neuron pacemaker behavior. Then, a small amount of noise can be added to the membrane potential. When the parameters are correctly tuned, the user saves the unit parameters and can print a copy of the graph, analogous to the one displayed in Fig. 1.1(a).

In PUM and BUM, the postsynaptic potential (PSP) parameters, namely their amplitude, rise and decay time constant, depend on the membrane on which the postsynaptic receptor is, and are consequently a neuron characteristic. The three parameters can be adjusted separately. In LIM, PSPs are modeled as pulses with a decay time constant identical to the membrane time constant.

A conditional burster is also available from the same neuron editor, for which parameters should be adjusted in order to get the dynamics models (an example is shown in Fig. 1.1(b)).



(a) The leaky integrator unit model (b) The bursting unit model

Figure 1.1: The phenomenological neuron models

In LIM, PSPs are modeled as instantaneous rising pulses with a decay time constant identical to the membrane time constant. On the contrary, For both, PUM and BUM, the post-synaptic potential (PSP) parameters, namely their amplitude, rise and decay time constant, depend on the membrane on which the postsynaptic receptor is, and are consequently a neuron property. The three parameters can be adjusted separately.

The conductance based model graphic editor

XNBC's conductance model is derived from the HH model, and incorporates 14 different transmembranar currents. This model explicitly takes into account the Na+, K+, Ca++, and Mg++ ions. The following currents are implemented by XNBC: I_{Na}, $I_{Na+\_persistant}$, I_{Ca}, I_t, I_K, I_M, I_A, I_{AHP}, I_C, I_H, I_{NMDA}, I_{leak}, I_{syn_epsp} and I_{syn_ipsp} (you can review these currents in Shepherd, ??). The currents are modeled using the dynamics of their activation/inactivation constants. The user adjusts each parameter –if desired values are different from defaults– by moving the double dials cursors (or by typing the value). The temporal evolution of the membrane potential and ionic currents change in real time according to the parameter values (Fig. 1.2).

The NMDA receptor for glutamatergic synapses neuromodulation, generic receptors, and a set of neurotransmitters are also implemented in this model (for a description of NMDA receptors in this model, see Vibert et al., 1994). Current and voltage clamp experiments can be simulated in order to adjust the conductance values. Drugs such as TTX and TEA can be released to compare the neuron behavior with and without the corresponding blocked channels.

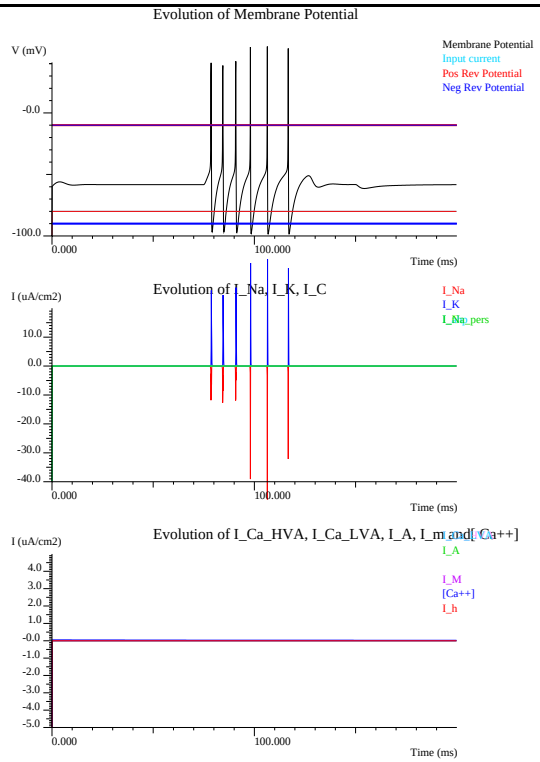


Figure 1.2: The conductance based model

A graphic submenu allows to plot any parameter versus any other parameter in order to study the dynamics of neuron behavior.

The postsynaptic potential (PSP) parameters, namely their amplitude, rise and decay time constants, depend on the membrane on which the postsynaptic receptor is, and consequently, are a neuron characteristic. The three parameters can be adjusted separately, and are modeled as a synaptic current I_{syn}, according to the chosen model.

The Virtual model

The neuron model can also be a “virtual model”, that is, an actually recorded neuron, whose spike time arrivals are used as input on the simulated neurons or networks. This allows to build hybrid networks.

Once the neurons are defined, they must be grouped together into networks.

1.3.2 Build a neural network

The connection matrix of the modeled network can be described using one of the two graphic network editors. According to the selected editor (simple or full, above the network editor pushbutton), the corresponding network editor is launched when the pushbutton is pressed.

The units of a single nucleus can be connected with the units of any other nucleus (including itself) through either all excitatory, all inhibitory or both excitatory and inhibitory synapses.

According to the type of network you want to build, the simple network editor can be enough, or the full featured network editor can be necessary. Figure 1.3 schematizes networks and the best editor choice.

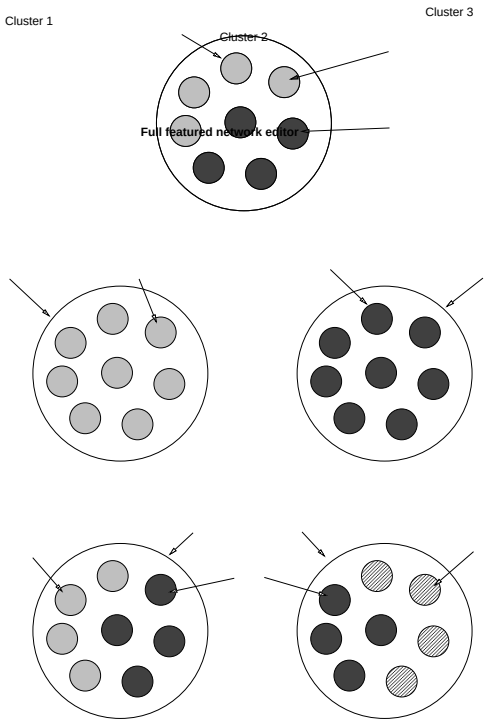


Figure 1.3: Networks and the network editor to use

The simple network editor

It allows to create nuclei with neurons that necessarily pertain to the same cluster. Nuclei based on different clusters (even for different neuron models, LIM, BUM, CBM or virtual) can be connected. It was designed for rapidly build networks made up of one or several nuclei. In addition, it allows to finely edit the connection matrix. Networks of several hundred of neurons are easily managed.

It does not allow to see individually the neurons inside the clusters, nor to geographically position the neurons nor to work in Horsley-Clarke coordinates, nor to describe the NMDA connections.

The full featured network editor

It allows to create nuclei containing neurons which can pertain to different clusters. It was designed for build small (or large if you have time...) networks, where nuclei have neurons from several clusters.

It allows also to finely edit the connection matrix and to choose the connection density around a given neuron, to see individually the neurons and their connections inside the nuclei, and to work in Horsley-Clarke coordinates. Connections with glutamate release acting on NMDA receptors can also be specified.

Once the network is built, the simulation can be run.

1.3.3 Prepare new drugs

It is possible to define a new arbitrarily named drug and which existing transmembranar current (one or several) it inactivates. Evidently, this is available only with CBM.

1.3.4 Run the simulation

Some questions concerning the name of the simulation, the iteration step, and the simulation duration, are asked when the simulator is launched. Once they are answered, the simulation may start. At present, the integration method used by the simulator is the exponential algorithm described in MacGregor (1987).

At any time, the simulation can be momentarily stopped in order to modify the external input to one or several nuclei, to give stimulation, drug, or change any parameter. It is also possible to modify some anatomical characteristics, such as connections between two nuclei, mimicking a lesion, or the membrane properties, mimicking pharmacological effects of drugs. The simulation can be stopped at any time, and the network state kept for later simulation.

During the simulation process, the network behavior can be observed on a graphic display representation if necessary. When the simulation is achieved, data may be saved into several file formats. These files are intended for the visualization tool, and the analyses tools, but also files for other graphic applications (e.g. gnuplot) are produced.

1.3.5 Visualize the simulation results

The visualization tool display the behavior of the modeled network with respect to time, using several representations with adjustable speed, color and time scale. It similar to a video tape player allowing to run, stop, go to a given iteration, or change the display speed. It allows to display:

- the joint color coded temporal evolution of the membrane potential of simulated units (and of units from a virtual cluster).
- spikes traveling along connections between neurons. This representation is convenient for studying the synchronization of spike trains.
- membrane potential recordings of selected units
- the global activity of the network.

1.3.6 Analyze the simulated network behavior

Two analysis tools are available:

- The time series analysis tool is mainly devoted to analyze the individual unit discharges (point process analysis). It offers 31 different analyses grouped into 8 submenus, such as instantaneous rate, interspike intervals, pre and post-stimulus histograms, Poincaré maps of intervals, auto and cross correlograms.
- The cluster activities analysis tool allows 8 different analyses such as FFT, Poincaré maps, amplitude correlograms, etc.

1.4 History and implementation

The first Neuro_bio_clusters (NBC), developed in 1988 on a MicroVax II, under Ultrix V2. Versions 2 and 3 of NBC, implemented a LIM and few analysis tools. Subsequently, beginning with version 4, NBC evolved toward a tool devoted to the general simulation of neural networks. In version 5 the CBM was introduced, and version 6 introduced the notion of virtual cluster, the graphic network editor and the visualization tool (Vibert et al., 1994b). NBC versions 4 to 6 were menu-driven, with only some parts using an *X Window* interface. Version 7 was the first version in which the control menu was replaced by an *X Window* interface, and was consequently renamed XNBC. With version 7 began also the possibility to mix the LIM and the CBM (Vibert et al., 1995b) within a single simulation. For XNBC V8, presented here, the interface was completely redesigned. That involves the second network editor, a new version of the ion-conductance based model editor, an *X Window* version of the simulator and the apparition of the concept of nucleus. XNBC development is still in progress through close collaboration with neuroscientists.

XNBC is written in portable ANSI C, and was compiled on Ultrix, Digital Unix, IBM AIX, SUN Solaris, HP Ux, Linux, DEC VMS and OpenVMS. XNBC runs on *X Window* workstations and needs the Motif library. When possible, the GNU C compiler (`gcc`) should be preferred. XNBC's data outputs are simple ASCII data files that can be easily converted to any format required by common graphic programs or spreadsheets. It produces native color PostScript files (that can be directly used to prepare figures).

XNBC is a public domain software package available freely for academic research purpose on Internet (`ftp://ftp.b3e.jussieu.fr/pub/XNBC`) and informations about new versions at URL `http://www.b3e.jus`

1.5 Contributors

The present version of XNBC is due to the joint efforts of many peoples, engineers, neurobiologists, biomathematicians, medical doctors, and all the users that asked for the interface enhancement, and that helped to discover bugs...

1.5.1 Project leader:

Dr Jean-Francois VIBERT, MD
B3E, ESI INSERM U444, ISARS
Faculté de Médecine Saint-Antoine
Université Pierre et Marie Curie (Paris 6)
PARIS, France.

1.5.2 Developers, in alphabetic order:

Below, the contributor and its affiliation, and the program they contributed to.

- **Evyatar Av-Ron**
  B3E, INSERM U444,
  `G_neuron, nbc_x`

- **Eric Boussard**
  B3E, INSERM U444, Universite Paris 6
  `G_neuron, xtms, xcaa, visu_nbc, network_editor, plot_xps`

- **Florence Cloppet**
  B3E, INSERM U263,
  `G_neuron`

- **Patrick Glandaz**
  B3E, INSERM U263,
  `link_edit`

- **Sylvain Hanneton**
  B3E, INSERM U263,
  `xcca, xtms, plot_ps`

- **Yann Herry**
  B3E, INSERM U263,
  `link_edit`

- **Denis Lambolez**
  B3E, INSERM U263
  `visu_nbc, G_neuron`

- **Khashayar Pakdaman**
  B3E, INSERM U444, Universite Paris 6
  `G_neuron, nbc_x`

- **Joël Pham**
  B3E, INSERM U444, Universite Paris 6
  `G_neuron, nbc_x`

- **Stéphane Rimbaud**
  B3E, INSERM U263
  `link_edit, network_editor, nbc_x`

- **Guy Sempé**
  DRET, Universite Paris 6
  `G_neuron`

- **Mike Stiber**
  Anatomy, UCLA, Los-Angeles, Ca, USA
  `P_neuron, nbc_x`

- **Marc Tisserand**
  DRET, Universite Paris 6
  `ihm, expert`

- **Jean-François Vibert**
  B3E, INSERM U444, Universite Paris 6, Paris
  `ihm, G_neuron, P_neuron, link_edit, network_editor, xtms, xcaa, visu_nbc, graph_nbc, pmb_cell_nbc, pmb_3d_nbc`

1.5.3 Neurobiologists, in alphabetic order:

- **Fabián Alvarez**
  Biofisica, Facultad de Ciencias, Montevideo, Uruguay

- **Nicolas Bourrié**
  B3E, INSERM U444, Université Paris 6, Paris and NAM, CNRS, Orsay

- **Jean Champagnat**
  Biologie Fonctionnelle du Neurone, IAF, CNRS, Gif/yvette

- **Jean-MArc Édeline**
  NAM, CNRS, Orsay

- **André Fabio Kohn**
  Escuela Polytechnica, USP, San-Paulo, Brazil

- **Efstratios K. Kosmidis**
  B3E, INSERM U444, Universite Paris 6, Paris

- **Vincent Leviel**
  Biologie Fonctionnelle du Neurone, IAF, CNRS, Gif/yvette

- **José-Pedro Segundo**
  Anatomy, UCLA, Los-Angeles, Ca, USA

- **John Stephens**
  Physiology, UCL, London, UK

- **Jean-François Vibert**
  B3E, INSERM U444, Universite Paris 6, Paris

We are pleased to thank all these peoples, as well as those that used the different versions and helped to discover bugs or gave ideas for the improvement of the user interface or the implementation of new features or related tools.

We are also pleased to thanks the DRET for it financial support (Contract 91/1246A and 94/2526A).

1.6 DISCLAIMER

XNBC8 IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND. WE MAKE NO WARRANTIES, EXPRESS OR IMPLIED, THAT IT IS FREE OF ERROR, OR IS CONSISTENT WITH ANY PARTICULAR STANDARD OF MERCHANTABILITY, OR THAT IT WILL MEET YOUR REQUIREMENTS FOR ANY PARTICULAR APPLICATION. IT SHOULD NOT BE RELIED ON FOR SOLVING A PROBLEM WHOSE INCORRECT SOLUTION COULD RESULT IN INJURY TO A PERSON OR LOSS OF PROPERTY. IF YOU DO USE IT IN SUCH A MANNER, IT IS AT YOUR OWN RISK. THE AUTHORS DISCLAIM ALL LIABILITY FOR DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES RESULTING FROM YOUR USE OF THE PROGRAM.

This software is provided as an Open Sources shareware.

It was developped for academic research purpose. Academic institutions and students can use it freely. Nevertheless, donations are welcome to help to maintain and develop XNBC. Non academic users are asked to contact the author to obtain the conditions for a commercial usage.

Chapter 2

The XNBC control panel

The control panel is displayed by the `xnbc` command that allows to control all the modeling process.

Many printed outputs are produced automatically by XNBC. It is wise, before to start XNBC, to define the convenient printer. This printer must be a PostScript printer. The printer name is controlled by the environment variable PRINT_PLOT_PS that should be positioned with the right print command and printer indicated.

For example in csh (or bsh or tcsh):

```
csh> setenv PRINT\_PLOT_PS "lpr -Plaser "
```

of with ksh (or sh)

```
ksh% PRINT_PLOT_PS="lpr -Plaser "
ksh% export PRINT_PLOT_PS
```

The XNBC control panel is run by typing at the Unix prompt:

```
csh> xnbc &
```

XNBC is made of a collection of programs launched by a control program displaying a user friendly control panel (Fig. 2.1). The control panel proposes several pushbuttons, each representing a task and launching this task when pressed. Pushbuttons are linked by arrows that symbolically represent the succession of tasks to run to perform a simulation.

A **Keep PostScript files** pushbutton allows to keep output PostScript files after they are printed (this button controls if graphic PostScript files are printed and deleted or only kept for further reprinting or use as a figure to include in a publication).

A **Browse the XNBC full manual** pushbutton allows to spawn a navigator (defined by the environment variable XNBC_BROWSER) to browse the user manual. The *Help Menu* allows to directly browse the Control Panel manual (help on control panel).

Simulations (here called Experiments) are assigned a name, chosen by the user. All files created during a given experiment are stored in a directory whose name starts when the experiment name and ends with `_nbc`. Keeping the same name as the basename of the directory, allows XNBC to open automatically the necessary files with asking questions. This name is kept in a file in the home directory in the file `.xnbcrc` in the home directory. This file records 4 important informations:

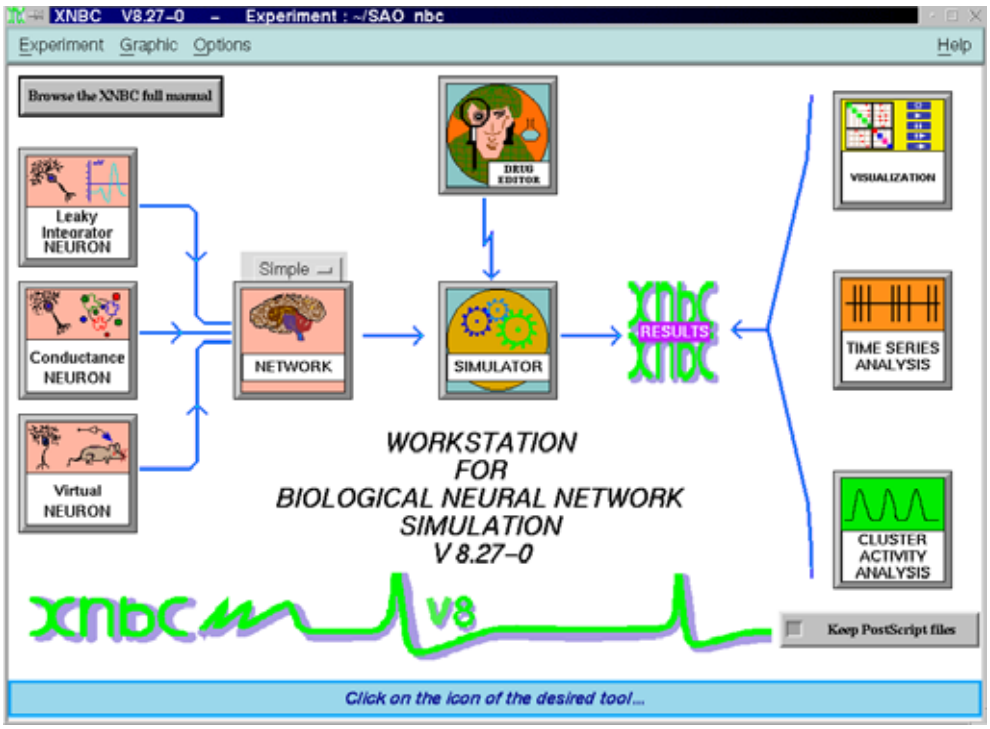


Figure 2.1: **The XNBC v8 control panel**

- the current simulation directory
- the current project name
- the current network name
- the current simulation name

This file is read by all components of xnbc to automatically propose the correct current worked file. Therefore it is not recommended to modify the experiment name during the experiment, since files will be no more automatically opened and will be asked to the user.

When moving the mouse, a dynamic one line short help text explains the pushbutton usage in the blue ribbon at the bottom of the screen.

2.1 Pulldown menus

The control panel proposes two pull-down menus (that can be teared off if the version of Motif is 1.2 or upper and if the tear off behavior is enabled in the Mwm personal control file). These menus allow to manage the simulation name (Experiment menu) and to call plotting programs (Graphic menu).

- **Experiment**

  Current: gives the name of the current experiment name and directory

  New: Creates a new directory to store files for a new experiment. This button opens a file selection box with the `*_nbc` directories already existing.

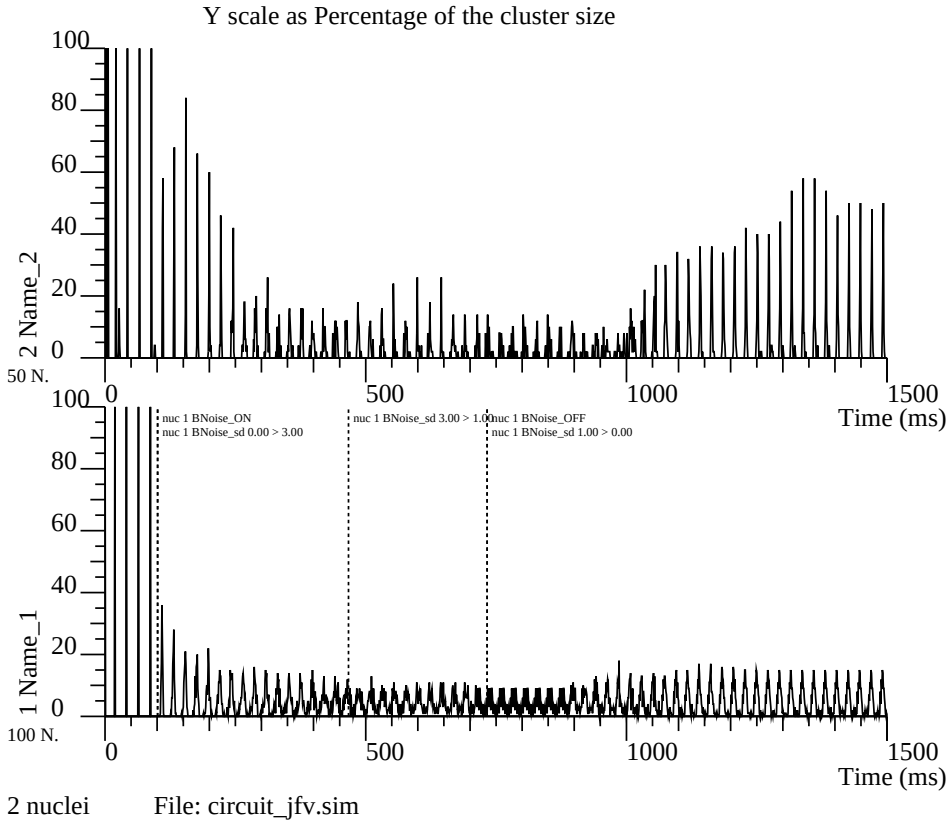


Figure 2.2: **Global activity of 2 nuclei obtained with the graphic tool**

Open: Open an existing directory to store files for an existing experiment. This button opens a file selection box with the `*_nbc` directories already existing.

Quit: exits xnbc.

- **Graphics**

Global activity of clusters: calls a program run in an xterm window. This program allows to plot the global activity of clusters or nuclei. Several representation options are proposed (Y axis units, length of X axis, control of what part of the experiment is plotted, number of pages, etc.). This program produces PostScript files, and is intended to prepare figures. When actions are took during the simulation (stimulation, parameter change, etc.), this is indicated on the graph. The same kind of plot can be obtained using the visualization program, but without the control provided by this program (Fig. 2.2).

Intracellular recording: calls a program run in an xterm window. This program allows to plot the membrane potential of selected units. Units are selected by giving their sequential number in the whole network (starting from 1): unit 1 is the unit 1 of nucleus (or cluster) 1. When all units are selected, type 0 to end the selection. Several representation options are proposed

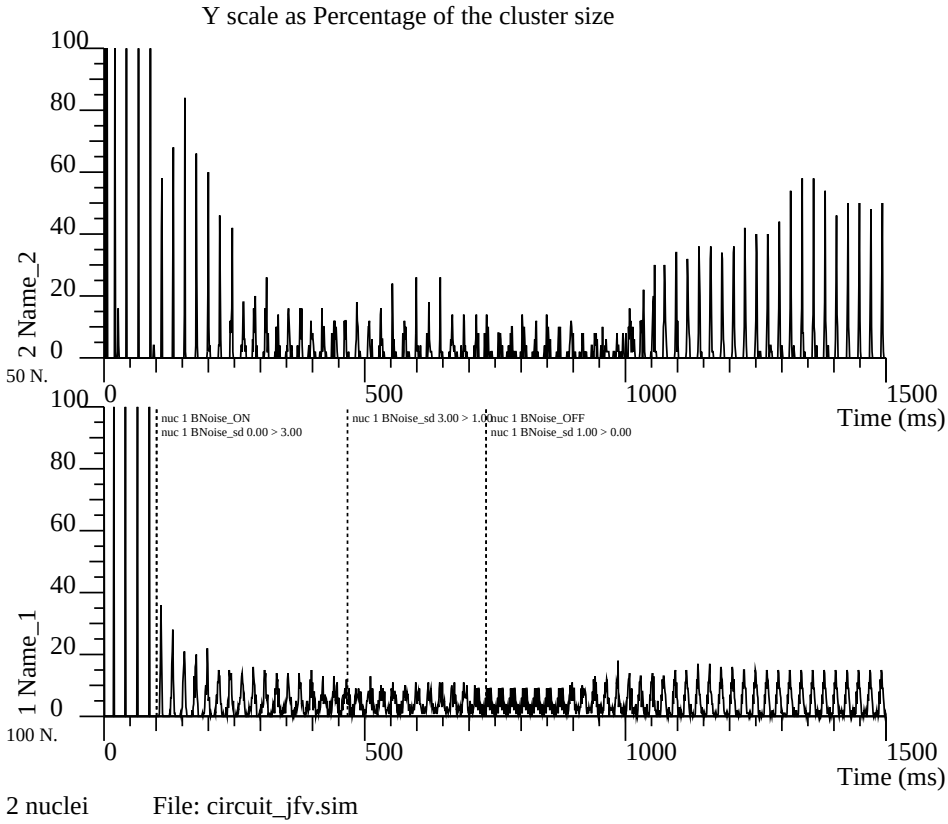


Figure 2.3: **Membrane potential of 3 units obtained with the graphic tool**

(Y axis size, length of X axis, number of pages, etc.). This program produces PostScript files, and is intended to prepare figures. The same kind of plot can be obtained using the visualization program, but without the control provided by this program (Fig. 2.3).

Membrane potential (3D): Same as the previous, but membrane potential are represented side by side, and can be viewed in a 3D representation whose rotations in X, Y and Z directions can be adjusted. This program produces PostScript files, and is intended to prepare figures (Fig. 2.4).

View PostScript Files: calls the PostScript previewer to preview the produced PostScript files by the simulation and analysis programs.

2.2 Programs launched by the control panel

The programs launched by the control panel are *a)* two graphic editors to adjust the neuron parameters (for the leaky integrator model (LIM) and for the conductance based model (CBM)), *b)* two network graphic editors (simple and fully featured), *c)* the simulator, *d)* the visualization tool and

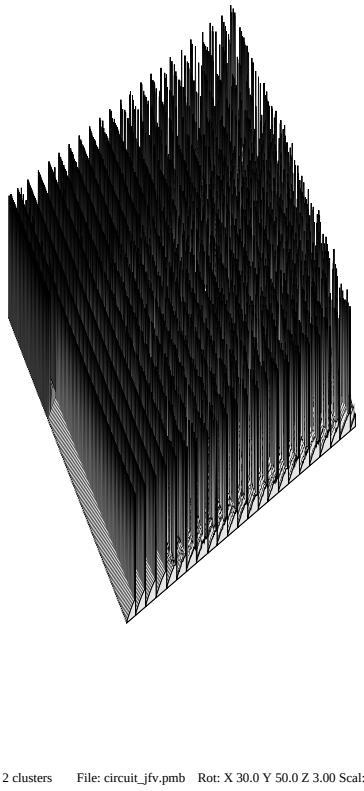


Figure 2.4: **3D representation of the activity of all units obtained with the graphic tool**

e) two analysis tools (time series and frequency analysis). The control panel displays pushbuttons arranged to guide the user to perform simulations and analyses. The user sequentially

- i) chooses the parameters of the neuron models,
- ii) builds a neural network
- iii) runs the simulation
- iv) visualizes the simulation results
- v) analyzes the simulated network behavior.

Nevertheless, the simulator itself (whose name is `nbc_x`, described in the corresponding chapter 8) allows to control also, in the same way, all the simulation process, using pull-down menus instead of graphic pushbuttons.

The pushbuttons launch the following tools. A separate chapter is devoted to each tool.

First *choose the parameters of the neuron models*. There exist two graphic editors to adjust the neuron parameters.

- **Leaky integrator NEURON** This pushbutton launches the phenomenologic neuron graphic editor to create LIM and BUM neuron types (or clusters). This tool is described in chapter 3.

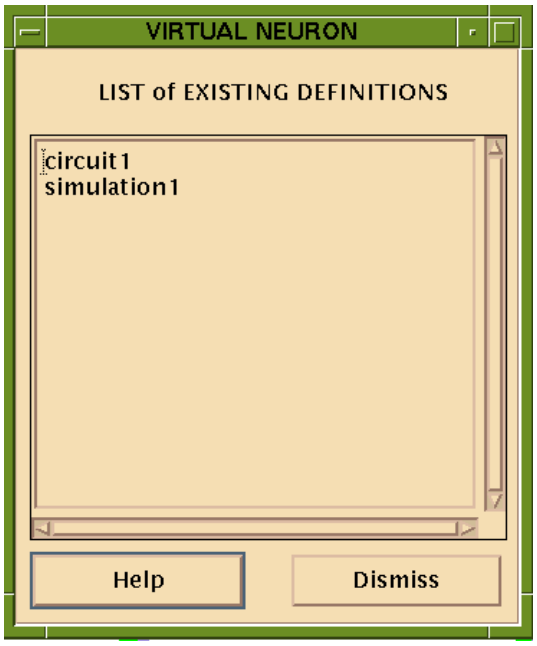


Figure 2.5: **The virtual cluster window**

- **Conductance NEURON** This pushbutton launches the conductance based neuron model graphic editor to create CBM neuron types (or clusters). This tool is described in chapter 4.
- **Virtual NEURON** The neuron model can also be a "virtual model", that is, an actually recorded neuron, whose spike time arrivals are used as input on the simulated neurons or networks. This allows to build hybrid networks. The pushbutton "Virtual cluster" gives a list of the available neuron files representing actual neurons (Fig. 2.5).

Once the neuron models are defined, neurons can be grouped together into networks, it is possible to *build a neural network*.

The connection matrix of the modeled network can be described using one of the two graphic network editors. According to the pushbutton selected above the network editor pushbutton (dark blue text), the simple or the full featured network editor is launched when the **NETWORK** pushbutton is pressed.

- **The simple network editor**

  It allows to build nuclei containing only one cluster each. It does not allow to see individually the neurons inside the clusters, nor to work in Horsley-Clarke coordinates. It does not allow to describe the NMDA connections. This tool is described in chapter 5.

- **The full featured network editor**

  It allows to build nuclei containing several clusters each. It allows to finely edit the connection matrix and to choose the connection density around a given neuron. Connections with glutamate release acting on NMDA receptors can also be specified. This is a rather sophisticated tool. This tool is described in chapter 6.

Once the network is built, the *simulation can be run* using the **SIMULATOR**pushbutton. This tool is described in chapter 8.

After the simulation, the user can

- *visualize the simulation results* by pressing the **VISUALIZATION** pushbutton. This tool is described in chapter 9.
- *analyze the simulated network behavior*. Two analysis tools are available:

  the **TIME SERIES ANALYSIS** pushbutton launches XTMS for time series analysis, and is mainly devoted to analyze the individual unit discharges (point process analysis). This tool is described in chapter 10.

  the **CLUSTER ACTIVITY ANALYSIS** pushbutton launches XCAA to analyze the nuclei activity. This tool is described in chapter 11.

2.3 Behavior of the control panel

When a task is launched and as long as it is active, the control panel pushbuttons become inactive (insensitive, indicated by the grey color of their text).

When a task is launched, a waiting message appears, and then automatically disappears after 10 s. If a task don't succeed, it is possible that this message stays displayed for few seconds before to be able to start again.

Sometimes, a window disappears from the screen, but the control panel stays insensitive. The window is probably hidden by the control panel. Move it, or lower it using the window manager top left menu.

2.4 Files

Input and output: `/.xnbcrc`

2.5 Diagnostics and problems

Self explaining diagnostics are provided on errors.

If a launched task stops abnormally, it is possible that the control panel stays insensitive, since it did not received the signal that the task has terminated normally. In this case, exit and rerun the control panel. Normally no data is lost, except those produced by the dead task. Only this one should be rerun.

Chapter 3

The phenomenologic model graphic editor

The phenomenologic model graphic editor is called `P_neuron`. P_neuron is an *X Window* graphic editor for the phenomenological neuron model of XNBC. It allows to choose the neuron parameters using sliding scales (on the right part and on the bottom of the screen) with visual feed back control in a graphic window displaying the unit membrane potential during an adjustable simulation duration (on the left part of the screen). Since many scales are available, they cannot be seen simultaneously on the screen. Scrollbars allows to shift the scales right and left (and up and down) to reach invisible scales.

Three EPSP and three IPSP can be positioned using scales and moved dynamically as scales are moved. The EPSP and IPSP weight are adjustable separately (but is the same for all three of each kind). They allow to make the neuron firing and to explore what happens when the weight changes (spike firing, IPSP, reversal potential approach, etc.). Equations are given in the Appendix C, Appendix B and Appendix D, as well as the default values.

3.1 The menu bar

A menu bar with pull-down menus allow to cope with files, parameters and options.

- **File**: allow to read, save, print the neuron, or quit P_neuron

  Read Neuron: reads back a previously saved neuron.

  Write Back Neuron: The parameter values are saved in a file with the .P_unit, .L_unit or a .B_unit extension according to the model worked on. This saves the file with the same as the previously read or saved file (if non name was given,This files can be read back by P_neuron (File-Read Neuron) and other XNBC programs. P_neuron is the default name.

  Save Neuron As...: The parameter values are saved in a file with the .P_unit, . or a .B_unit extension according to the value of the Conditional Burster option. This save the file with a new name that is asked using a file selection box. This files can be read back by P_neuron (File-Read Neuron) and other XNBC programs. P_neuron is the default name.

  Print: produces a PostScript hard copy file of the neuron membrane potential evolution, automatically spooled on the printer indicated by the `PRINT_PLOT_PS` environment variable. The

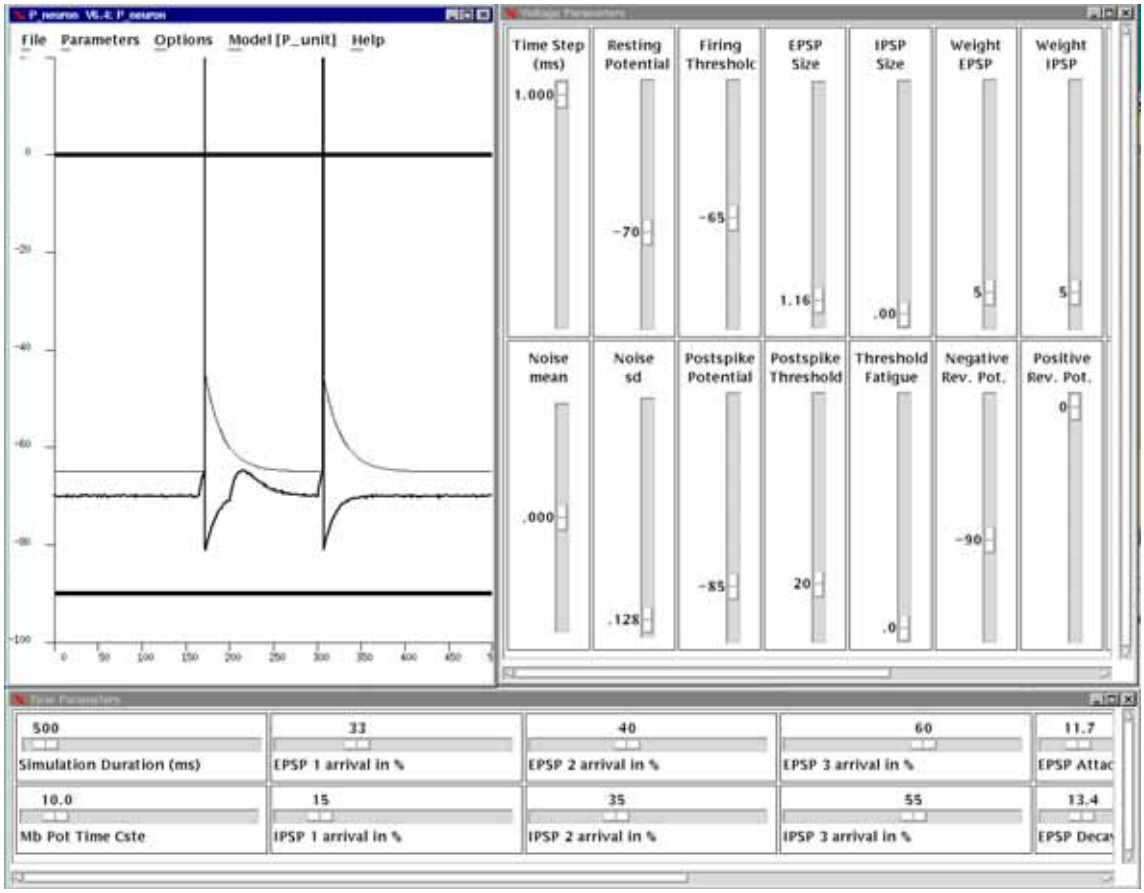


Figure 3.1: **The P_neuron user interface**

output file file_in.P_unit.ps, file_in.L_unit.ps or file_in.B_unit.ps is spooled on the PostScript printer, and then deleted if the **Keep PostScript files** is not set on the control panel.

Quit: exit the phenomenologic editor `P_neuron`.

- A menu bar **Parameter** item proposes several buttons:

  Change Parameter: call back to front screen the two panels with scales (if they were previously closed).

  Redraw: redraws the neuron (if any X server problem).

  Back to defaults: reload all the default parameters for scales. Scales are positioned back to these values.

- A menu bar **Options** item proposes several buttons:

  Kohn colored noise: Makes the noise filtered using an algorithm described by Andre Fabio Kohn (USP, San Paulo, Brazil). The filtering is controlled by the Resistance scale, activated when this option is set.

  Static Update: When moving the scales positioning the 3 EPSP or the 3 IPSP, the neuron graph is updated dynamically. On slow or overloaded computers, this can be disturbing. Setting this option disable the dynamic update of these scales.

Less Fatigue: The adaptation of the neuron is controlled through the threshold increase after each spike. If no fatigue is set, the threshold increase is constant. This is controlled using the fatigue threshold scale. If fatigue is set, the threshold increase increases, proportionally to the scale value. Less Fatigue option modifies the algorithm used to modify the fatigue.

Scales: allows to modify the vertical scale of the membrane potential representation.

- A menu bar **Model** item proposes a radiobox set of buttons:

  Phenomenologic: allows to simulate the PUM model When this option is set, files read and written have the extension `.P_unit`.

  Leaky Integrator: allows to simulate the LIM model When this option is set, files read and written have the extension `.L_unit`.

  Conditional Burster: allows to simulate a conditional burster (BUM), with two threshold: one threshold fires normal spike, while the other -generally above the first- starts a burst that stops only if an IPSP arrives, or if the predefined duration is over. The intra burst firing frequency is adjusted using scales, as well as a composition factor, a multiplying factor applied on the firing frequency when an EPSP arrives during a burst. The corresponding scales are activated when this option is set. When this option is set, files read and written have the extension `.B_unit`.

- A menu bar **Help** item allows to browse the P_neuron manual.

3.2 The vertical scales

The following vertical scales are always active and available:

- Resting Potential
- Post-spike Potential
- Firing Threshold: the usual firing threshold
- Post-spike Threshold: controls the unit adaptation
- EPSP Maximum
- EPSP Weight: number of synaptic buttons, acts as a multiplier of EPSP maximum.
- IPSP Maximum
- IPSP Weight: number of synaptic buttons, acts as a multiplier of IPSP maximum.
- Positive Reversal Potential: EPSP reversal potential
- Negative Reversal Potential: IPSP reversal potential
- Noise mean: mean of the Gaussian noise added to the membrane. Acts as a constant input if noise sd is null.

- Noise standard deviation: noise amplitude, centered on the membrane potential value.

The following scales are active when the Conditional Burster option is set:

- Spike Frequency: intra burst base frequency (when the first EPSP crosses the burst threshold)
- Max Frequency: Maximum intra burst frequency (this scale is linked to the preceding, and moves alone when moving the preceding because the scale minimum is the spike frequency value)
- Composition Factor: the multiplying factor when a second EPSP arrives (see above).
- Burst Threshold: second threshold, that triggers a burst
- Post-spike Burst Potential: value of the membrane potential after a burst
- Min Burst Duration
- Max Burst Duration: Burst duration is chosen randomly between these two values.

The following scale is active when the Noise Filter option is set:

- Filter Resistance: value of the filter resistance: decreasing the resistance decreases the filter time constant (more high pass band); whereas decreasing the resistance shifts the filter toward low pass band.

3.3 The horizontal scales

The following horizontal scales are always active and available:

- Simulation duration: in ms (maximum: 10 seconds)
- Membrane Potential Time Constant
- Arrival time of EPSP 1: in percentage of the simulation duration
- Arrival time of EPSP 2
- Arrival time of EPSP 3
- Arrival time of IPSP 1
- Arrival time of IPSP 2
- Arrival time of IPSP 3
- Threshold Time Constant exponentially from 0 to 1 after the spike. All neuron inputs are multiplied by this factor.

The following scales are active for the PUM and the BUM

- EPSP Attack Time Constant: time constant of the increasing part of the EPSP
- EPSP Decay Time Constant: time constant of the decreasing part of the EPSP
- IPSP Attack Time Constant: time constant of the decreasing part of the IPSP
- IPSP Decay Time Constant: time constant of the increasing part of the IPSP
- Shunt Time Constant: time constant of a factor increasing exponentially from 0 to 1 after the spike. All neuron inputs are multiplied by this factor.

3.3.1 PostScript outputs

P_neuron produces a PostScript file `file_name_P_unit.ps` that prints the membrane potential. It is automatically spooled and deleted. Files can be kept if the Keep PostScript pushbutton of the XNBC control panel is set.

3.3.2 Saving neuron

The parameters of the neuron can be saved in a file whose extension is .P_unit or .B_unit. The file produced can be read back by P_neuron (and related programs, as XNBC V8 [7]).

To read a neuron, choose in the menu bar the [File] popdown menu, and the [Read neuron] pushbutton. A file selection box will pop up to give a name. No extension is needed, it is added to the name automatically. If changing parameters, further savings can be done using the [Write neuron back...] pushbutton in the [File] popdown menu. It is a prudent habit to save regularly his work. If no file was read previously, and thus no name was given previously, the file selection box will pop up to give a name.

To save a neuron, choose in the menu bar the [File] popdown menu the [Save neuron as...] pushbutton. A file selection box will pop up to give a name. No extension is needed, it is added to the name automatically. When a name has been given, the [OK] pushbutton saves the neuron. Consequently, further savings can be done using the [Write neuron back...] pushbutton in the [File] popdown menu. It is a prudent habit to save regularly his work.

When exiting, and if a parameter has been changed after the last saving operation, a message warns the user that the file was not saved, and allows to return to the program to normally save the neuron.

3.4 Files

Input files: `file_in.P_unit, file_in.L_unit, file_in.B_unit`

Output files: `file_in.P_unit, file_in.L_unit, file_in.B_unit, file_in.P_unit.ps, file_in.L_unit.ps, file_in.B_unit.ps`.

3.5 Known problems

Self explaining diagnostics are provided on errors.

Chapter 4

The conductance based model graphic editor

The conductance based model graphic editor is called $\mathbf{G\_neuron}$. G_neuron is intended to simulate a single neuron that can receive some connections from other neurons at chosen times and with chosen synaptic weight. G is here to recall the conductance electrical symbol, since the model implements the transmembranar currents using the Hodgkin-Huxley formalism[2]. The user acts on the model behavior by interactively adjusting the current conductances (G).

The current version (see About G_neuron in the Menu Help) implements 12 different transmembranar currents plus 2 synaptic currents.

The currents are modeled using a simplified HH type of kinetics (e.g. see Av-Ron et al., 1991 [1]) and Michaelis-Menton type of kinetics for chemical activation/inactivation variables. For a description of this model, see Vibert et al., 1994b[6].

G_neuron allows to simulate a conductance based model (CBM) of neuron based on the Hodgkin-Huxley (HH) model (Hodgkin and Huxley, 1952 [2]). The CBM incorporates 14 different transmembrane currents. This model explicitly takes into account the Na^+, K^+, Ca^{++}, and Mg^{++} ion concentrations. The following currents are implemented in the CBM: I_{Na}, $I_{Na^+{}_{persistant}}$, I_{Ca}, I_t, I_K, I_M, I_A, I_{AHP}, I_C, I_H, I_{NMDA} and I_{leak}. A new current can be added at the C source level. The source is documented to make it relatively simple[1]. The currents are modeled using a simplified HH type of kinetics and Michaelis-Menton type of kinetics for chemical activation/inactivation variables (Av-Ron et al., 1991). The neuron editor is used to adjust the parameters of the CBM including all the ion-kinetic parameters. Noise may be added to the membrane potential. Once the parameters are tuned, they can be saved for further use.

Three integration methods are proposed: exponential (MacGregor, 1987 [3]), Euler and order 4 Runge-Kutta. All parameters of the CBM can be individually adjusted. The user interface of G_neuron is user friendly. Parameters are adjusted by moving the double dial cursors (or by typing the value), while the temporal evolutions of the membrane potential and ionic current (left part of the screen)

[1] Each place where something has to be added or modified is flagged by `#ifdef ADD_NEW_CURRENT` and a pattern is given, that must be duplicated below the `#endif`, modified according to the new current following the example given (generally, only the variable names has to be updated –I_New must be replaced by I_My_new_current–) and evidently, the formulas to compute it...

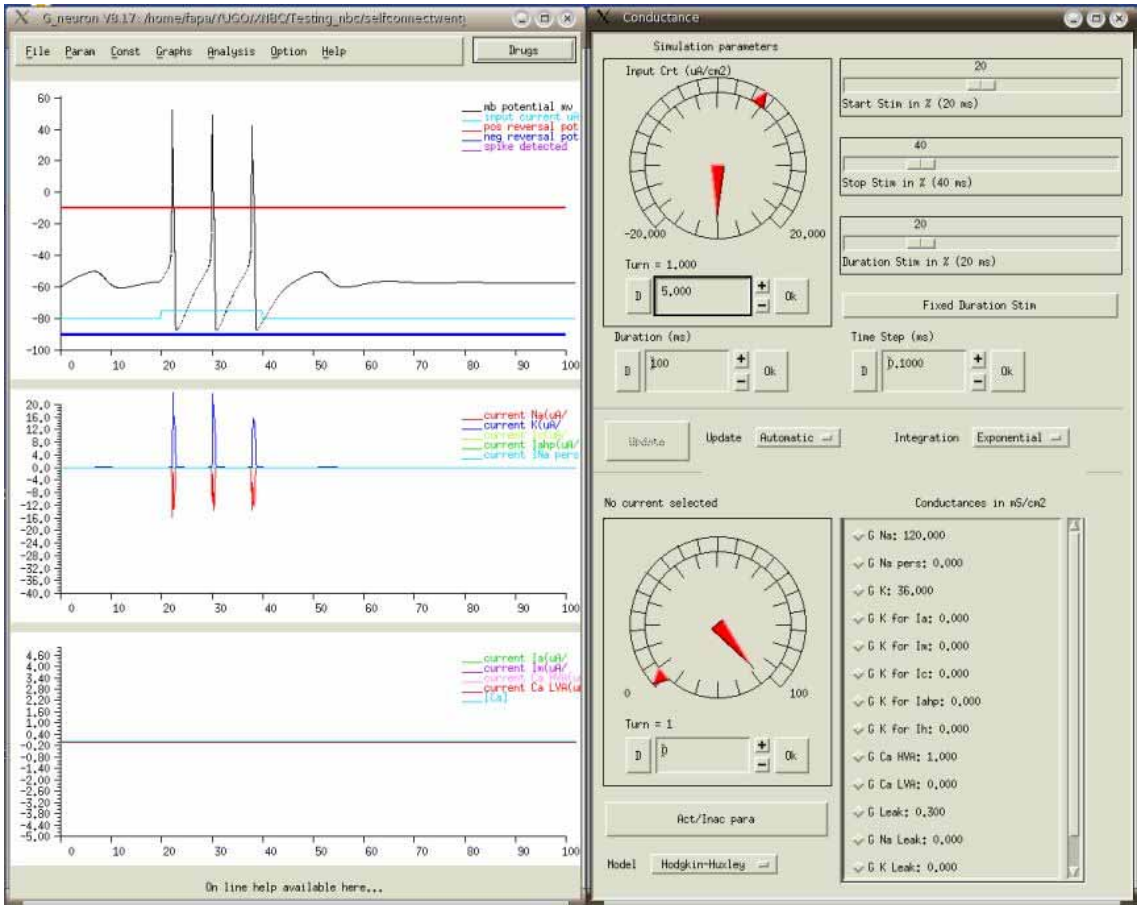


Figure 4.1: **The G_neuron user interface.** Left: the modeled neuron (upper part: potential, two lower parts: currents; right: dials and input windows used to adjust the parameters. Times in ms.

change in real time according to the updated parameter values. A graphic submenu allows to plot any variable versus any other variable. Current and voltage clamp experiments can be simulated in order to adjust the conductance values.

Effect of drugs such as TTX and TEA can be simulated to compare the model behavior with and without the corresponding blocked channels. The postsynaptic potential are modeled using alpha functions with rise and decay time constant and amplitude as one of the two synaptic currents (I_syn_epsp and I_syn_ipsp). Synapses with glutamate release can induce an NMDA neuromodulation with a long lasting membrane modification.

Printout are produced as PostScript files automatically spooled on the printer indicated by the environment variable PRINT_PLOT_PS (see Output Help topic and section 3.3.1).

4.1 User interface

The man-machine interface allowing data input and interaction with the program is done through a user friendly *X Window*/Motif graphic user interface. All parameters have reasonable default values that are always at a mouse click.

Model parameters are adjusted with scale cursors or dials with an on line dynamic visual feedback on the neuron membrane potential shape and the current temporal evolution. Figure 4.1 displays the graphic interface for G_neuron. On the left part, the modeled neuron is drawn (upper part:

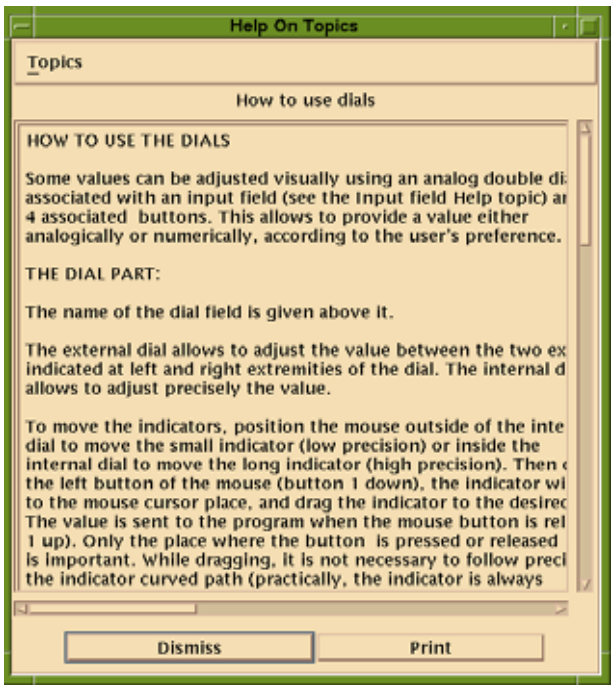


Figure 4.2: **Help window of G_neuron**

potential, two lower parts: currents); on the right part, dials and input windows are used to adjust the parameters. Times are in ms. Neuron parameters can be saved to constitute a neuron data base, since the conductance parameters, the time constants and many other parameters can greatly vary among neuron categories.

On line help is available for most of objects on the screen, at the bottom of the left panel. More detailed help (in fact the different parts of this manual are available form the [Help][On Context] pulldown menu of the main menu bar (Fig. 4.2). Topics can be selected from the pulldown [Topics] menu. Help can be printed on the the printer defined by the `PRINT_PLOT_PS` environment variable (see PostScript output section).

4.2 Currents

Fourteen different currents can be used and separately adjusted.

4.2.1 Current selection

All transmembranar currents can have their parameters adjusted separately. Since they necessitates many parameters to describe a current, these parameters are grouped in several windows.

The activation/inactivation parameters are grouped together, the equilibrium potential too, and only the conductance adjustment (lower dial on the right panel) and the current selection are directly available to the user. The radio box panel at the right of the lower dial allows to select a current. The dial acts only after a current as been chosen and selected by double clicking on one of the radio buttons. The radio box is in a scrolled window, and, according to the screen size, it can be necessary to move the scrolling bar to get access to the lower ones. When a current is selected,

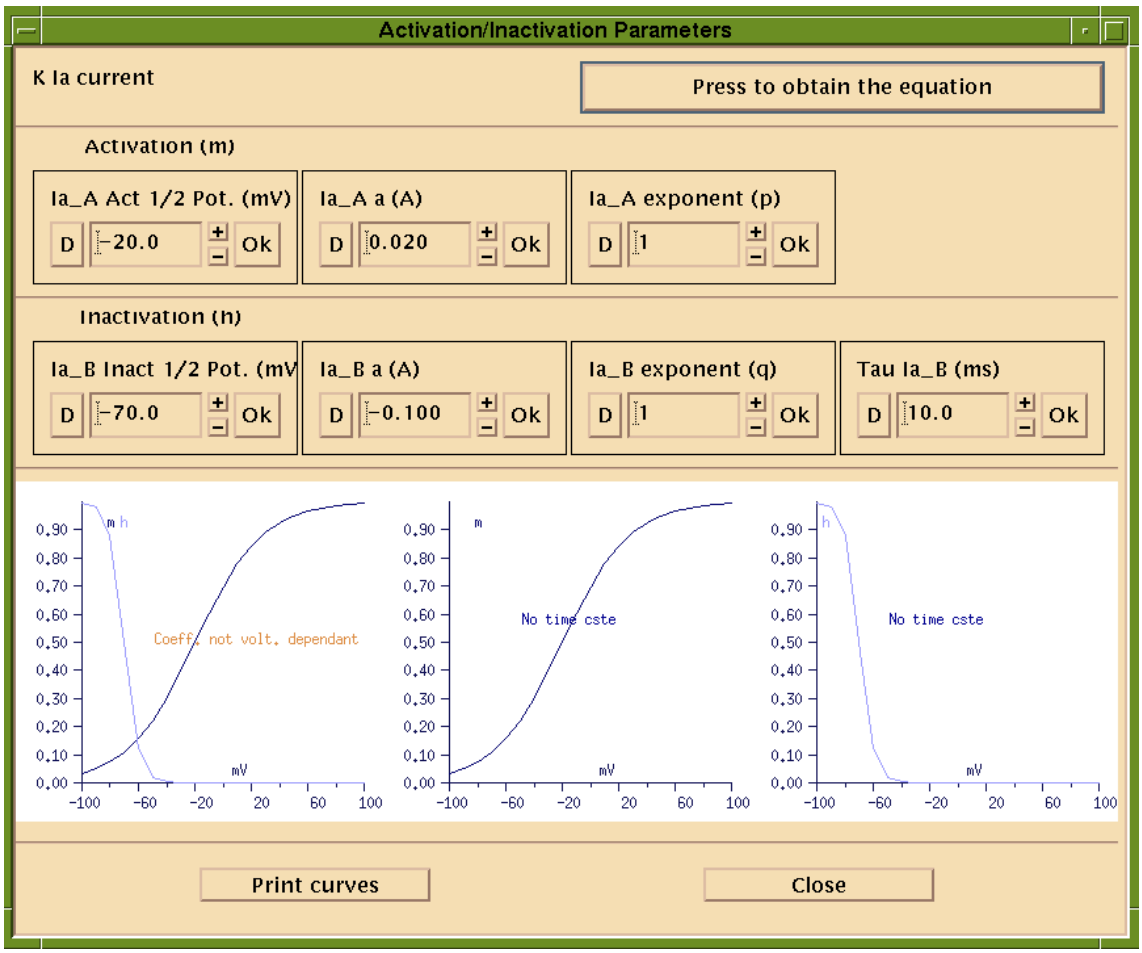


Figure 4.3: **Panel for current parameters' adjusting**

its name appears above the lower dial, and the current conductance value appears in the input text field below the dial (see How to use dials Help topic and section 4.7.3). When pressing the [Activation/Inactivation parameters] pushbutton, the correspondent panel is displayed (see Current parameters Help topic and section 4.2.1).

4.2.2 Equations used

All equations used to make the simulation can be obtained by pressing the [Equation] pushbutton located on the panel used to adjust current parameters (see Current parameters (section 4.2.3) and NMDA (section 4.2.4) Help topics). Equations are given in the Appendix E.

4.2.3 Currents parameters

Current activation/inactivation parameters are adjusted using a general panel of 8 input text field (see How to use input fields (section 4.7.4) Help topic)(Fig. 4.3). The 4 input text field upper row are for activation, the 4 input text field lower row for inactivation. Only some of the input fields can be presents according to the selected current. At the bottom of the panel, three graphs are visible where the evolution of the relevant parameters versus membrane potential is represented. Modifying the parameter values automatically updates the corresponding graphs.

The [Print curves] pushbutton at the bottom of the panel produces a color PostScript file (*G_unit_activ.ps) to keep a hardcopy of the graphs. Parameter values are printed together with the graph. The [Press to obtain the equation] pushbutton at the top of the panel pops up the equation used to simulate the selected current action.

The panel proposes the following parameters:

- [V 1/2 Activation (m)] and [V 1/2 Inactivation (h)]: The voltage for which the sigmoid has its inflexion point. This parameter shifts the sigmoid toward left (low value) or right (high value).
- [A coefficient for m] and [A coefficient for h]: The slope of the sigmoid. This parameters change the sigmoid slope. Low values give low slopes, high values steep slopes.
- [Exponent for m (p)] and [exponent for h (q)]: Corresponds to the number of particles in the HH equations
- [Lambda for m] [Lambda for h]: Parameter that control the maximal Tau for a current. Corresponds to a coefficient related to the temperature.
- [Tau for m] [Tau for h]: A time constant for activation or inactivation that is needed for some currents.

4.2.4 NMDA current

The NMDA current requires parameters different than the other currents. This is why its parameters are grouped together in a separated panel (Fig. 4.4).

Selecting the NMDA current is done either as for other currents, or using the [NMDA] item in the [Parameters] pulldown menu of the menu bar. Since the NMDA synapses are special one, specific glutamate releasing synapse have to be installed on the soma to see the effect of the glutamate release on the NMDA receptors. This is why the NMDA panel proposes the possibility to connect NMDA synapses to the soma. The number, the time arrival and the synaptic weight can be adjusted separately for each connection (see Neuronal connections Help topic and section 4.4.3). The other parameters are:

- the time constant of increasing slope of the NMDA EPSP
- the time constant of decreasing slope of the NMDA EPSP
- the Eta parameter η: a coefficient linking the NMDA channels conductance to the Mg++ concentration
- the Gamma parameter γ: coefficient of the voltage dependence of the NMDA channels
- the reversal potential for NMDA synapse
- the extracellular concentration of Magnesium

An [Equation] pushbutton at the bottom of the panel pops up the equation used to simulate the NMDA action.

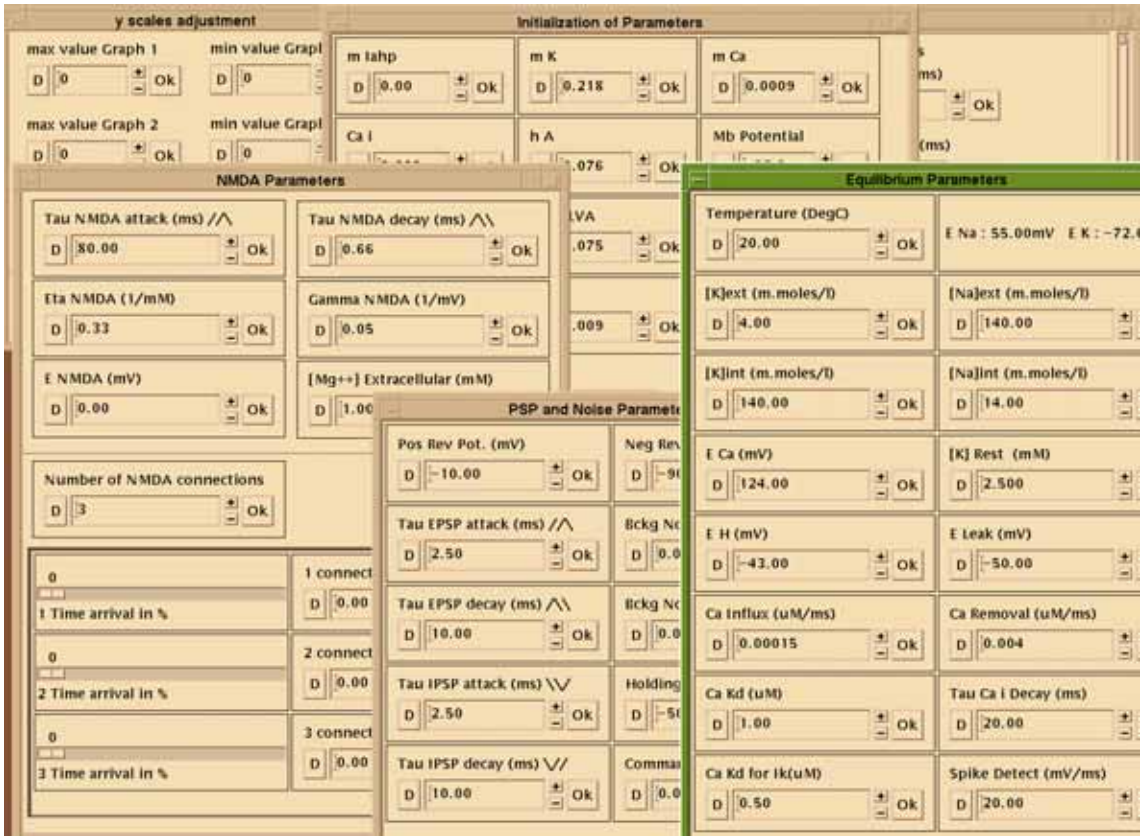


Figure 4.4: **Panels to adjust most of G_neuron parameters**

4.2.5 Initial values

G_neuron set all variables to default initial values. These values are used each time the simulation run (automatic or manual update) (see Manual versus automatic update Help topic and section 4.3.1). These values can be modified by the user. Some constants such as equilibrium potentials can also be changed (Fig. 4.5).

Choose in the [Constants] popdown menu the [Initializations] item. This opens a panel with 12 input text fields, and choose the value to modify (see How to use input text fields Help topic).

For the equilibrium potentials modification, choose in the [Parameters] popdown menu the [Equil. pot., sp. det...] item. This opens a panel with 14 input text fields, among them several concern equilibrium potentials and others ion concentrations. Choose the value to modify (see How to use input text fields Help topic (section 4.7.4).

4.3 Simulation parameters

The simulation can be controlled by several parameters. Time step and duration, integration method, manual or automatic updating, can be chosen (see Fig. 4.1).

4.3.1 Manual versus automatic update

Each time a parameter is modified, the simulation is run and the graphs are updated. When long simulations and very small time step are required, this can be a serious waste of time. This is why

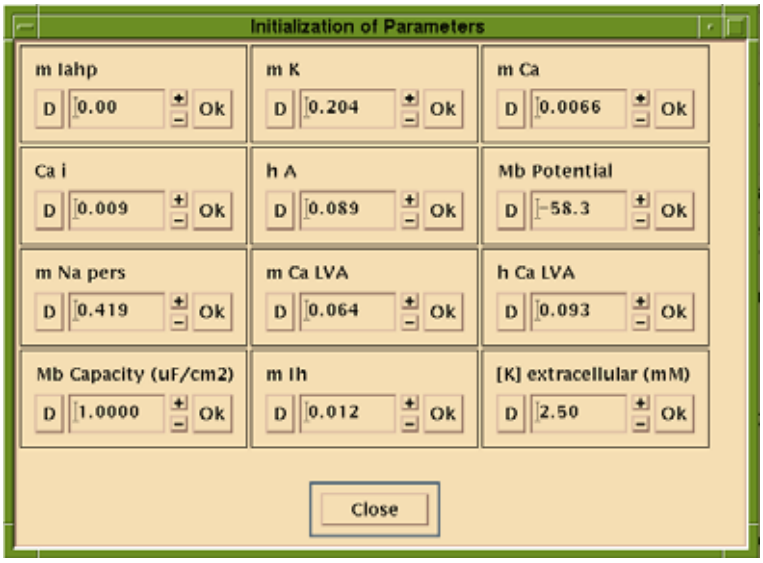


Figure 4.5: **Panels to adjust initial values of G_neuron parameters**

it is recommended to modify all the necessary parameters and only then re-evaluate the simulation and update the graphs.

This is the reason of the option menu labeled [Update] and proposing [Automatic] (default) and [Manual] options. When [Manual] option is set, the [Update] pushbutton is set sensitive and the computation and graphics are updated only when pressing it. In this case, if the window size is changed or a window moved, the graphs are not updated, and become blank until the next time [Update] is depressed. When [Automatic] (default) option is set, the [Update] pushbutton is set insensitive and the computation and graphics are updated each time an [OK] button (or Return hit) is done in an input field, or a dial modified, or again a window resize or move is performed.

4.3.2 Simulation times

Two input text fields (see How to use input fields Help topic) allow to set the simulation duration (defaulting to 100 ms) and the time step (defaulting to 0.1 ms). Both values can be changed at will. The left panel of the screen shows the result of the simulation using graphics (the temporal evolution of the membrane potential and the transmembranar currents). The size of the panel can be modified to expand the graph, in any direction. All other panels are scrolled windows that automatically add scroll bars when necessary to cope with small screens.

4.3.3 Integration methods

Three integration methods are implemented in this version of G_neuron.

The exponential integration method (see Mac Gregor, Brain modeling, 1987 [3]), the default method, the Euler method and the order 4 Runge-Kutta method (from "Numerical Recipes in C", Version 2.02 [5]). The first method is more rapid, but can be less accurate when precise spike counts are required. The Euler method is more accurate, but needs smaller time steps. Prefer the more accurate (and slower) Runge-Kutta method. Choice between methods is done using the option menu labeled [Integration] and proposing [Exponential] (default), [Euler] and [Runge-Kutta] options. The

current selected integration method is those currently appearing on the option menu. Note that the Euler method is valid only if the time step is equal to or less than 0.01 ms. When the Euler method is selected, the time step is reset to 0.01 ms if necessary. A message panel warns the user. Note also that the Runge-Kutta method is valid only if the time step is equal to or less than 0.025 ms. When the Runge-Kutta method is selected, the time step is reset to 0.025 ms if necessary. A message panel warns the user.

4.4 Experiment type

Several types of experiment can be performed: voltage or current clamp, external constant input, external neuronal inputs, background noise modification, drug action.

4.4.1 Voltage clamp

G_neuron allows to make simulated voltage clamp experiments where Command potential and Holding potentials can be adjusted (see Fig. 4.4).

First, choose in the menu bar the [Parameters] popdown menu and the [Simulation] item. A window will pop up to select the experiment type: Current or Voltage clamp. The default choice is the Current clamp. Select the Voltage clamp radio button, and close the window ([Close] button at the bottom of the window). Then choose in the same [Parameters] popdown menu the [PSP, noise, ...] item. The two Holding and Command potentials parameters are in this panel. The holding voltage sets the value of an imposed new "resting" membrane potential. The command potential is the potential value added to the holding potential and monitored to keep the voltage constant at the clamped value (clamped potential = holding_potential + command potential). The command potential start and stop is controlled by the start stimulation and stimulation duration input fields (see Stimulation control Help topic (section 4.4.2)). Voltage clamp experiment are only available with Exponential integration method

4.4.2 Stimulation control

Stimulation can be given to the neuron. Stimulation starts at the time given in the [Start Stimulation at (ms)] scale, and its end is given in the [Stop Stimulation at (ms)] scale. Its duration is given in the [Stimulation duration (ms)] scale field. A push button allows to keep the duration fixed and to move the stimulation moment by either the beginning or the end. Scales are updated according. Its amplitude is adjusted interactively using the upper dial in the Conductance window on the right part of the screen. The stimulus is materialized on the upper graph panel by a sky blue line, below the membrane potential.

4.4.3 EPSP and IPSP

To see the effect of the excitatory and inhibitory synaptic conductance, it is necessary to connect other neurons to the simulated neuron. PSP can be added, or removed or modified at will. One of

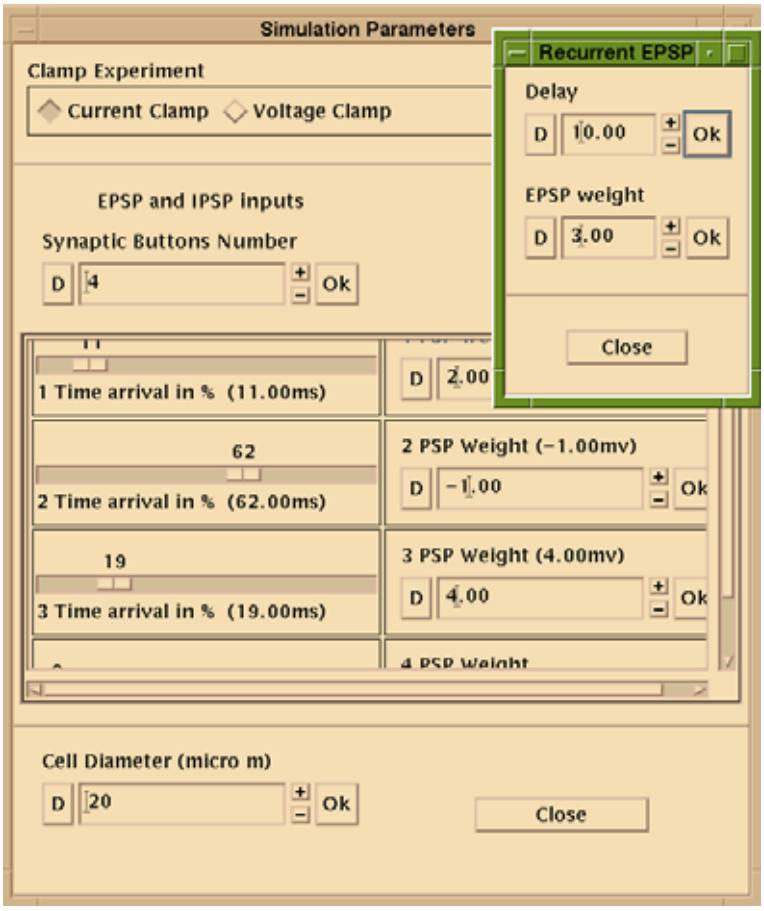


Figure 4.6: **Panel to adjust PSP inputs in G_neuron**

the panels allows to adjust the post synaptic current characteristics: time constant of the first part, called PSP attack symbolized by a double line (increasing for an EPSP, decreasing for an IPSP), time constant of the second part called PSP decay (decreasing for an EPSP, increasing for an IPSP). Time constants are in ms (Fig. 4.4).

Choose in the [Parameters] popdown menu the [PSP, noise, ...] item. The left part of the panel proposes 4 input text fields to adjust these 4 parameters (see How to use input fields Help topic). To adapt the number of synaptic buttons, choose in the [Parameters] popdown menu the [Simulation] item. Choose in the [Parameters] popdown menu the [Simulation] item. A panel pops up and allows to select the synaptic button characteristics (Fig. 4.6). The middle part of the panel proposes one input text fields to choose the number of synaptic buttons to create. When validate a corresponding set of one scale and one input field text appears in the window below the input field.

The scale at left allows to position the EPSP at the correct time of arrival, while the scale at right allows to chose the psp weight. Positive number gives EPSP while negative gives IPSP.

It is recommended to chose first the weight, and then to adjust the time of occurrence using the scale and looking at the graph. If the menu item [Option/Static update] is not selected, the graph is updated dynamically, in real time by the scale. If [Option/Static update] is selected, the update is done only when releasing the mouse button. This is necessary when computation is long (long simulation, short time step, or Runge-Kutta integration method).

4.4.4 Recurrent PSPs on the simulated neuron

It is possible to send back an EPSP or an IPSP to the neuron after a spike fired. The option menu has a recurrent item [Option/Recurrent] allowing enable or disable this feature. If enable, a panel pops up to give the weight of the EPSP and the delay (in ms) between the firing and the EPSP.

Spike detection must be enabled to detect firing and send recurrent psp. If not, the recurrent button has no effect.

4.4.5 Hybrid simulations

Hybrid simulations are simulation were input to the simulated neuron comes from experimental data. These one must be stored in a [.tms] file (those produced for xtms, see the chapter 10 for file format and xtms usage). The [File] menu proposes an item to select an input .tms file in which the arrival time of EPSP are stored. EPSP or IPSP time occurrence are read from the selected file.

4.4.6 Background noise

Background noise can be added to the membrane potential to mimic the synaptic noise. The noise added is a Gaussian noise whose mean and standard deviation can be chosen by the user. Choose in the [Parameters] popdown menu the [PSP, noise, ...] item. The two background noise mean and standard deviation (sd) parameters are in this panel (Fig. 4.4).

No noise is obtained by setting both parameters to 0 (their default values). Increasing the noise mean provides a constant input. Increasing the noise sd increases the noise amplitude. Generally, only the sd parameter is changed, with a mean noise of 0 (centered on the membrane potential value).

4.4.7 Drugs

Drugs can be added to the simulated neurons. TEA that blocks the Potassium channels and TTX that blocks the Sodium channels are implemented. Practically, they simultaneously block all the concerned channels (by transiently setting to zero the corresponding conductances). Other drugs can be simulated by setting to zero the corresponding conductance(s). See the Trends in Neurosciences Supplement, TINS, 1996 [4]) to know what current is affected by what drug. A [Drugs] pushbutton at the rightmost part of the menu bar pops up a window with one option menu per drug. If set to [Yes], the drug in effect is activated and deactivated if set to [No].

4.5 Visualizations

Several graphic representations are available to visually control the simulation process and the parameters adjustments.

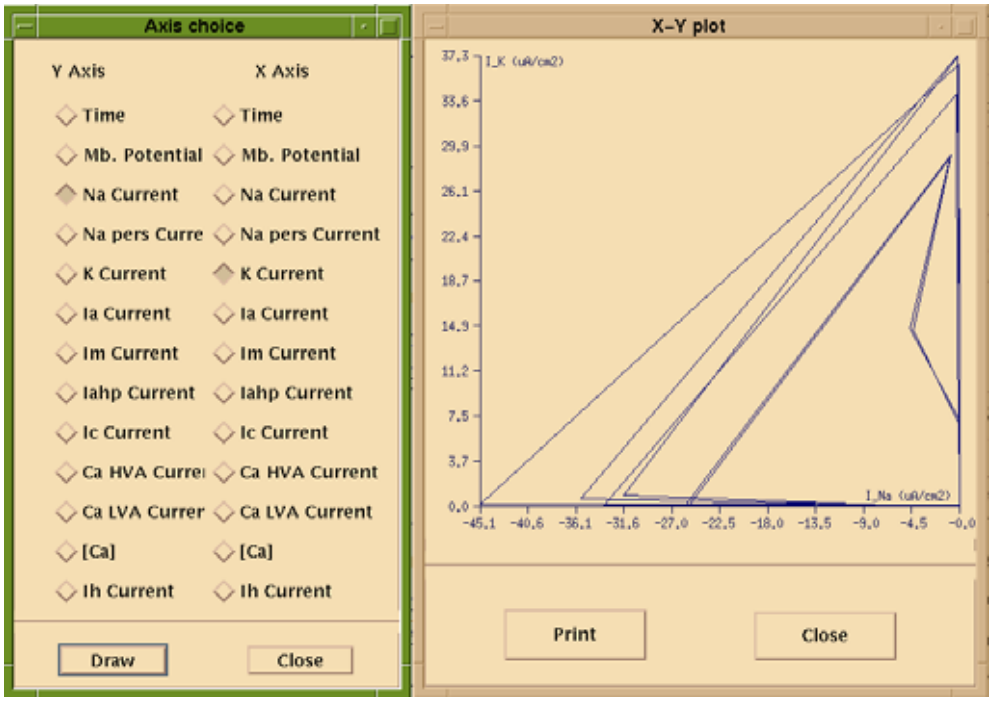


Figure 4.7: **Panel to select X and Y axis and phase plane representation**

4.5.1 Temporal representations

G_neuron displays the temporal evolution of many relevant variables (membrane potential, currents, stimulus) on the left panel of the screen. These representations are permanently visible on the left part of the screen. The screen is divided onto three panels. The top panel is for membrane potential versus time representation. Simultaneously, are represented the equilibrium potential of Na^+ and K^+, the EPSP and IPSP reversal potential as well as the external input. The middle and bottom panels are for the different currents, plus the $[Ca^{++}]$ (see Fig. 4.1). Axis scales can be modified using the [Parameters Menu/Scales...]. A window appears where the 3 axis are adjustable separately for minimum and maximum.

4.5.2 Phase plane representations

Other graphs can be obtained. Any variable can be plotted versus any other variable, in order to study the relationship between any couple of parameter. When the parameters are chosen, a specific window displays the graph (Fig. 4.7) . This a two step process: first, choose what to put on X axis and what to put on Y axis and second, draw the graph. The graph can be saved and plotted as a PostScript file (see Outputs Help topic).

Choose in the [Graphs] popdown menu the [Axis choice] item. This opens a panel with a double list of all variables available for drawing. Select one in each column (Y: left column; X right column) and press the [Draw] pushbutton. Once this choice is done, the graphs can be obtained directly from the [Draw] item of the [Graphs] popdown menu. The panel with the graph appears then. It can be resized at will to modify the drawing appearance. To obtain a PostScript hardcopy, press the [Print] pushbutton. The window can be rescaled at will in order to choose the best representation. A pushbutton allows to produce a PostScript hardcopy of the plotted graph. The produced plot keeps

the proportions set by the user by recalling the window. The printout will automatically select the way it is printed (portrait or landscape) in order to fit correctly on the page.

4.5.3 Activation/inactivation versus voltage representations

During the adjustment of the current parameters (see Current parameters help), the corresponding graph of the activation and inactivation versus voltage are displayed at the bottom of the window (three graphs) (see Fig. 4.3). These graphs are dynamically updated when the value of a parameter is modified. A pushbutton allows to produce a PostScript hardcopy of the displayed graphs.

4.6 Outputs

Several types of outputs can be obtained: a neuron description file can be saved (and read back later), as well as PostScript files representing the membrane potential and the different currents as a function of time, or one as a function of the other. Currents' values can also be saved for further analysis.

4.6.1 PostScript outputs

G_neuron provides several types of graphs on screen. All can be obtained as PostScript files and printed for hard copies using the plot_ps package. They are automatically spooled and deleted. Files can be kept if the Keep PostScript pushbutton of the XNBC control panel is set.

The following graphs can be obtained:

- A copy of the 3 panels with the potential and the currents,
- A copy of the X-Y plot of nay variable versus any other variable,
- A copy of the currents parameters curves (activation, inactivation curves). The produced PostScript files are name using the following conventions:

  `file_name_G_unit.ps`: the membrane potential and currents

  `file_name_G_unit_XY.ps`: the XY phase plane plot

  `file_name_G_unit_activ.ps`: the current activity/inactivity curves

4.6.2 Saving time series

G_neuron allows to save in a file the spikes point process (the date of arrival of spikes). This allows to:

- perform time series analysis of the spike train generated by the current simulation (the values stored in the file are always those resulting from the simulation visible on screen). From the Analysis menu, it is possible to launch the xtms time series analyses (the xtms program). See the specific manual for xtms usage. The file has a [.tms] extension.

- feed the simulated neuron by a spike train either previously simulated or produced from experimental data.

The [File] menu proposes an item to select an output tms file name.

Spike detection must be enabled to detect firing and store spike times. If not, the button has no effect.

4.6.3 Saving neuron

The parameters of the neuron can be saved in a file whose extension is .G_unit. The file produced can be read back by G_neuron (and related programs, as XNBC V8 [7]) to continue the simulation later.

To read a neuron, choose in the menu bar the [File] popdown menu, and the [Read neuron] pushbutton. A file selection box will pop up to give a name. No extension is needed, it is added to the name automatically. If changing parameters, further savings can be done using the [Write neuron back...] pushbutton in the [File] popdown menu. It is a prudent habit to save regularly his work. If no file was read previously, and thus no name was given previously, the file selection box will pop up to give a name. No extension is needed, it is added to the name automatically.

To save a neuron, choose in the menu bar the [File] popdown menu the [Save neuron as...] pushbutton. A file selection box will pop up to give a name. No extension is needed, it is added to the name automatically. When a name has been given, the [OK] pushbutton saves the neuron. Consequently, further savings can be done using the [Write neuron back...] pushbutton in the [File] popdown menu. It is a prudent habit to save regularly his work.

When exiting, and if a parameter has been changed after the last saving operation, a message warns the user that the file was not saved, and allows to return to the program to normally save the neuron.

4.7 Miscellaneous

4.7.1 Spike detection

This feature is intended for the people using G_neuron as a neuron editor for the XNBC simulator ([7]. It allows to detect the presence of spikes using a slope criteria. This feature is necessary to produce the time series (*.tms files) to analyze the neurons behavior, and is necessary since the conductance based model has no firing threshold as it is found in the leaky integrator model. Choose in the [Parameters] popdown menu the [Equil. pot., sp. det...] item. The spike detection input field is at the bottom of this panel (see Fig. 4.4). Adjust the value in order to get a vertical purple line at the very beginning of the each spike. The value is not very critical, the parameter being robust. Normally the value is between 10 and 30 (default: 20). Spike detection can be disabled by setting a setting a large value or by using the [Option/Spike detection] menu.

This menu and its submenu allow to be either disabled or enabled. When enable the visible marker (the purple vertical line) is drawn on the potential graph. This can be disturbing, thus the marker can be itself set invisible or not in the same submenu.

Spike detection is necessary if time series output (see section 4.6.2) and recurrent stimulation.

4.7.2 Help

Help is provided either on line at the bottom of the left panel, or using the [Help] menu allowing to browse the manual or to read the different topics directly in, without the figures.

4.7.3 How to use the dials

Some values can be adjusted visually using an analog double dial, associated with an input field (see the Input field Help topic) and its 4 associated buttons. This allows to provide a value either analogically or numerically, according to the user's preference.

The dial part

The name of the dial field is given above it. The external dial allows to adjust the value between the two extrema indicated at left and right extremities of the dial. The internal dial allows to adjust precisely the value.

To move the indicators, position the mouse outside of the internal dial to move the small indicator (low precision) or inside the internal dial to move the long indicator (high precision). Then click the left button of the mouse (button 1 down), the indicator will jump to the mouse cursor place, and drag the indicator to the desired position. The value is sent to the program when the mouse button is released (button 1 up). Only the place where the button is pressed or released is important. While dragging, it is not necessary to follow precisely the indicator curved path (practically, the indicator is always on the radius made by the mouse cursor and the dial center).

While dragging the indicator, the value indicated in the text field below the double dial is updated in real time, allowing to precisely control the desired value.

When reaching either the maximum or minimum, the dial sounds the bell and indicates above the input text field that the limit is reached and that you have to turn back.

The text field input part:

As an alternative, the value can be entered by typing it directly in the editable text field associated to the dial. When a value is precisely known, this can be a more rapid and precise way to enter a value.

The left pushbutton labeled D (for Default) recalls the default value associated with this field. Generally, this is the value indicated in the field when the program starts.

The right pushbutton labeled OK validates the input and makes it available to the program. Usually (but it is dependant on the *X Window*s configuration colors), when a new value is typed in the text field, the name of the input field (above it) is highlighted until the OK pushbutton is pressed or the RETURN key hit.

The two small pushbuttons labeled + and - allow to respectively increase or decrease the value. Each time the + pushbutton (or the -) is pressed the rightmost digit is increased (or decreased) by one.

Pressing these pushbuttons continuously change the value slowly at the beginning and accelerates exponentially.

To enter the value manually, simply select the editable text field an type in the value. Usually (but it is dependant on the *X Window*s configuration colors), when a new value is typed in the text field, the name of the input field (above it) is highlighted until the OK pushbutton is pressed or the RETURN key hit.

Only a fixed number of decimals is allowed for a given input field. If you try to enter more, a bip will sound. If an integer is asked, a bip will sound if you type a decimal point. In any case all characters except the set [0123456789.e+-] will sound the bip (as well as more than one . or sign).

Adjustment of the default value and of the number of decimals

The default value is predefined when the program starts, as well as the number of digits after the decimal point. These values can be modified interactively by pressing simultaneously the shift key and one of the mouse buttons while the mouse cursor is in the dial area part of the dial input widget:

- Shift Left Button: takes the current value as the new default.
- Shift Middle Button: decreases the digits number after the decimal point.
- Shift Right Button: increases the digits number after the decimal point.

Note, if the current value has no decimals, the new default will don't have decimals. Take care of putting the right number of decimals before to change the default value.

4.7.4 How to use the input fields

Each input is given using a specific input field[2] and the associated 4 buttons.

The name of the input field is given above the editable text field. The left pushbutton labeled D (for Default) recalls the default value associated with this field. Generally, this is the value indicated in the field when the program starts.

The right pushbutton labeled OK validates the input and makes it available to the program. Usually (but it is dependant on the *X Window*s configuration colors), when a new value is typed in the text field, the name of the input field (above it) is highlighted until the OK pushbutton is pressed or the RETURN key hit.

The two small pushbuttons labeled + and - allow to respectively increase or decrease the value. Each time the + pushbutton (or the -) is pressed the rightmost digit is increased (or decreased) by one. Pressing these pushbuttons continuously change the value slowly at the beginning and accelerates exponentially.

To enter the value manually, simply select the editable text field an type in the value. Usually (but it is dependant on the *X Window*s configuration colors), when a new value is typed in the text field, the name of the input field (above it) is highlighted until the OK pushbutton is pressed or the RETURN key hit.

[2] Authors: Denis Lambolez B3E, INSERM U444 Paris, France.

Only a fixed number of decimals is allowed for a given input field. If you try to enter more, a bip will sound. If an integer is asked, a bip will sound if you type a decimal point. In any case all characters except the set [0123456789.e+-] will sound the bip (as well as more than one . or sign).

4.8 Files

Input: `file_name.G_Unit`

Output: `file_name.G_unit`, `file_name_G_unit.ps`, `file_name_G_unit_XY.ps`, `file_name_G_unit_activ.ps`

4.9 Known problems

Voltage clamp is available only with exponential integration. No other known problems.

Chapter 5

The simple network editor

The name of the simple network editor is `link_edit`. The simple network editor allows to create networks were nucleus and clusters are equivalent. No distinction is possible between between them.

This tool is accessible from the XNBC control panel via the “Simple network editor” pushbutton. It is used to graphically edit simple networks and to create the files necessary to run the simulation: the “.len”, “.wgt”, “.clu” “.anat” and “.lnk” files:

- “.lnk” file is the link configuration file that contains all the parameters of the network architecture: number of nuclei (or clusters), link between them and inside them, and so on. It is used by the simulator and the visualization tool.
- “.len” file contains the axon lengths
- “.wgt” file contains the synaptic weights
- “.clu” file contains the cluster names used by this network.
- “.anat” file contains the network anatomy, that is so far not used, but will be useful in the next versions.

5.1 The user interface

When called, the simple network editor presents a blank window with menus in which clusters, neurons and links will be drawn (Fig: 5.1).

- Menu File

  Save : save all files (.len, .wgt, .anat, .lnk) of the current network.

  Save As : idem, but you can give another file name.

  Open : open an existing .lnk file to modify the network.

  New : create a new network (noname.lnk by default).

  Exit : exit link_edit.

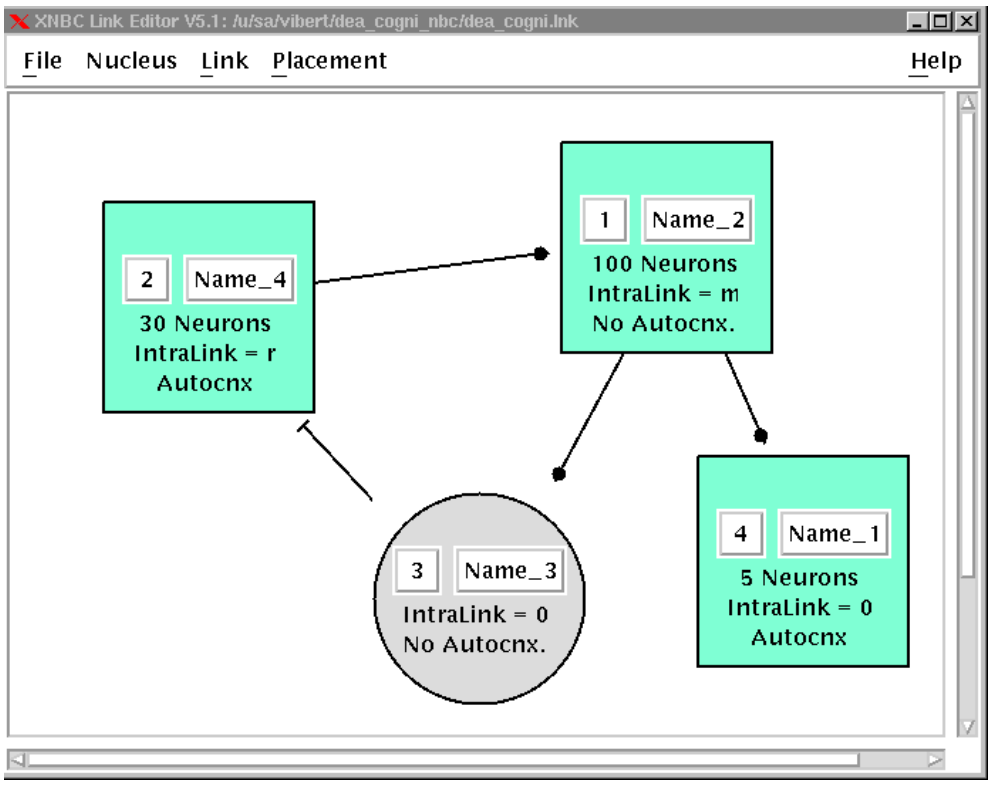


Figure 5.1: **The simple network editor editor interface**

- Menu Cluster

  New : create a new cluster for the current network.

  Copy : duplicate an existing cluster (same parameters).

  Delete : delete an existing cluster.

  Modify : modify an existing cluster.

  Swap : swap two existing clusters (in automatic placement only).

  Move : move an existing cluster (in manual placement only).

- Menu Link

  New : create a new link between two existing clusters or to an existing cluster on itself.

  Delete : delete an existing link.

  Modify : modify an existing link.

- Menu Placement

  Automatic : automatic placement of clusters in the window.

  Manual : manual placement of clusters in the window.

  Remarks:

  1. These two menus are exclusive. A star after the menu indicates which one is selected.

  2. Placement in the window does not' t reflect anatomic coordinates of the cluster. It s just to allow a good organization of clusters in the window.

3. In the automatic mode, clusters are placed on a circle in the order of their creation. In the manual mode, clusters can be manually positioned using the mouse, in a way that reflects the network structure.

- Menu Help

  On Context : Choose a topic

  Browse manual: self explaining.

5.2 The cluster

A cluster (or nucleus) is represented by a square with two buttons and informations.

- Informations

  Horsley-Clarke coordinates of the center of the cluster.

  IntraClust indicates if the cluster is self-connected. In the case of a self-connection, the sign of the connection (- means Inhibitory connection, + means Excitatory connection) is indicated, else there is a zero.

- Buttons

  the button with the number of the cluster is used to select the cluster for different actions (modify it, swap with an other cluster, create link, etc... see below).

  the button with the name of the cluster is used to see or modify cluster parameters. This button activates the Cluster Parameters dialog box (see below).

Remark:
In manual placement mode, to move a cluster, click in the cluster, maintain the button, drag the cluster to the wanted location then release the button. The network is redrawn.

5.3 The link

A link is represented by a line from the first cluster to the second. A link is like a nerve. It contains all neuron axons of the first cluster. A symbol indicates which cluster of the twice is the receiver. This symbol indicates too the kind of connexion:

- A point represent an excitatory connection.
- A bar represent an inhibitory connection.
- A bar plus a point represent an random connection. Some synapses are excitatory, other are inhibitory.

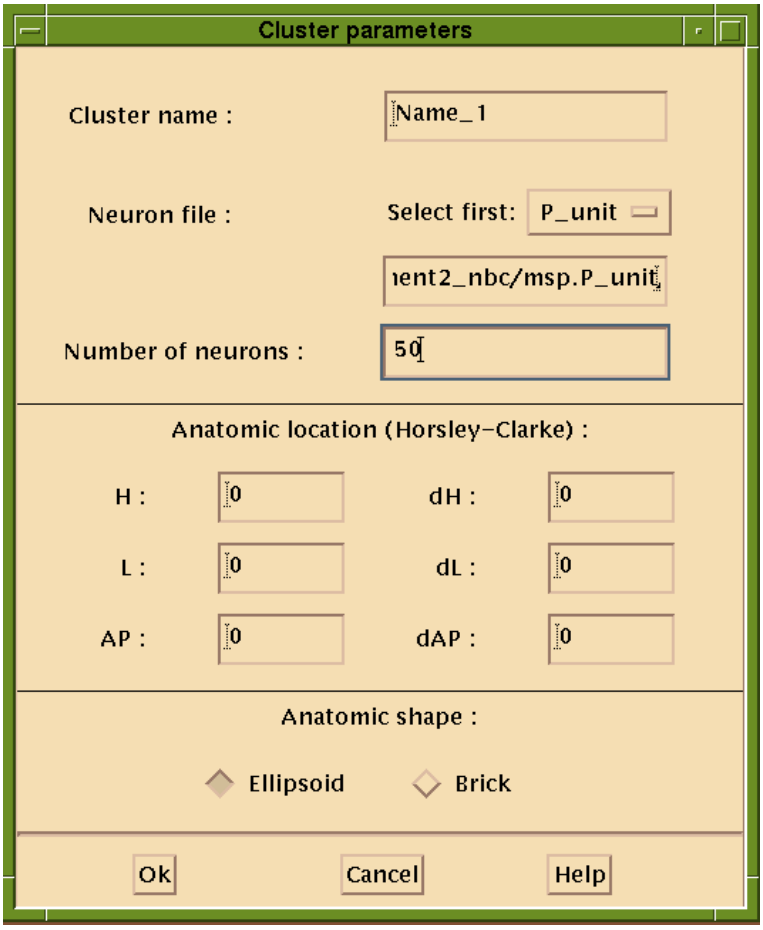


Figure 5.2: **The Cluster Parameters dialog box of the simple network editor**

5.4 The Cluster Parameters dialog box

This box appears with different actions:

- with a click on the button with the name of the cluster,
- when select the Cluster/New menu,
- when select the Cluster/Modify menu.

This box (Fig. 5.2) allows to enter parameters of a cluster (a new one or an existing one):

- Name of the cluster: enter a symbolic name in the text field (default name is Name_number of the cluster.
- Neuron file: select first a model type. This will open a file selection box to choose a *.?_unit file where are defined neurons parameters. Or enter the file name in the text field if the correct model type is already selected.
- Number of neurons: enter in the text field the number of neurons for the cluster.
- Anatomic location: Horsley-Clarke coordinates of the cluster. In the left column enter location of the center of the cluster, and in the right column enter size of the global envelope of the cluster.

- Anatomic shape: choose the shape of the global envelope of the cluster. These choice are exclusive.

Click on OK to create or modify the cluster or on Cancel to cancel creation or modification.

5.5 The Link Parameters dialog box

This box appears with different actions:

- when select the Link/New menu,
- when select the Link/Delete menu.
- when select the Link/Modify menu.

In each case, after have select the menu, click on the button with the number of the first cluster then click on the button with the number of the second cluster. First and second cluster can be the same in the case of a self-connection.

This box (Fig. 5.3)allows to enter parameters of a link (a new one or an existing one):

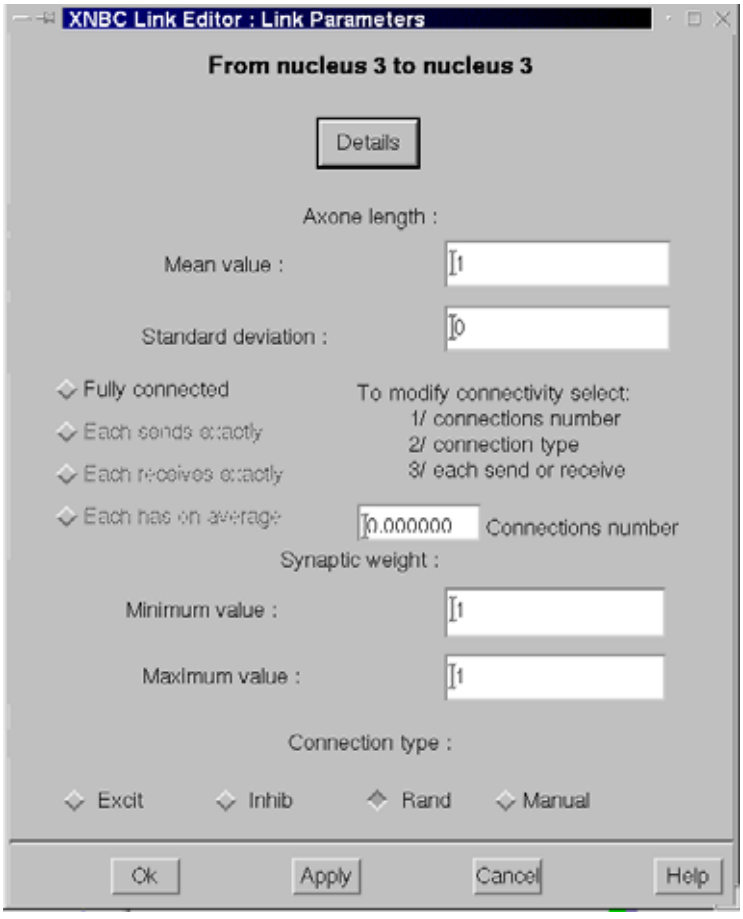


Figure 5.3: **The Link Parameters dialog box of the simple network editor**

- Axon length: enter mean value and standard deviation of axons length (Gaussian repartition).

- Synaptic knobs number: reflects the weight of connections. Enter the minimum and the maximum values (Gaussian repartition).
- Connection type: choose the type of connection: Excitatory, Inhibitory, Random (Excitatory + Inhibitory with a random repartition) or manual. These choice are exclusive.

In the case of manual connection type, connections neuron to neuron can be set by clicking on the Details button (see known bugs).

Click on OK to create or modify the Link or on Cancel to cancel creation or modification.

5.6 The steps to follow to create a simple network

A Open an existing .lnk file or create a new one.

B Create a new cluster or modify an existing one.

Remark: To select a cluster click on the button with its number.

C Set cluster placement: to swap two cluster in automatic mode, select Cluster/Swap menu then select two clusters. In manual placement mode drag cluster to the wanted location.

D Create a new link or modify an existing one.

E Save files

5.7 User Interface Menu Functions.

- The file menu.

  New : clears the current network, edit a new one.

  Open... : lets the user choose a network to be opened.

  Save : saves all files for the current network.

  Save as... : saves all files with a new name.

  Exit : simply exits the program.

- The cluster menu.

  New : creates a new cluster for the current network.

  Copy : duplicates the next cluster you select.

  Delete : deletes the next cluster you select.

  Modify : modifies the next cluster you select.

  Swap : swaps the two next clusters you select.

  Move : lets you move the next cluster you click on.

- The link menu.

  New : creates a new link between the two next clusters you select (the same cluster can be selected twice.

  Delete : deletes the link between the two next clusters you select.

  Modify : modifies the link between the two next clusters you select.

- The placement menu.

  Automatic : sets the automatic placement mode of the clusters on the work area: they are placed on a circle.

  Manual : sets the manual placement mode: the user clicks on the cluster he wants to move, this one follows the mouse pointer as long as the mouse button is pushed.

- The help menu.

  On context : with different topics

  Browse manual: self explaining.

5.8 Representation of objects

5.8.1 Cluster representation.

A cluster (or nucleus) is represented by a square with two buttons and informations, such as the Horsley Clarke coordinates of the center of the cluster, a flag that indicates if the cluster is self-connected, if it is, then the nature of this self-connection, ń + ż meaning excitation, ń - ż inhibition, ń r ż randomized, ń 0 ż no self-connection.

The button with the number of the cluster is used to select the cluster for different actions such as modifying it, swapping it with an other cluster,and so on.

The button with the name of the cluster activates the cluster parameters dialog box, offering the user the ability to modify the clicked cluster.

5.8.2 Cluster and Neuron Distinction.

To distinguish a neuron from a cluster, the program draw a circular gray cluster when it is only one neuron, and green square when it has more than one neuron.

5.8.3 Link representation.

A link is represented by a line from the first cluster to the second. It is like a nerve. It contains all axons of first cluster neurons. A symbol indicates which cluster of the both connected is the receiver. This symbol indicates the kind of the connection too:

- point represents an excitation connection
- segment represents an inhibition connection

- point + segment represents a randomized connection

5.9 Placement mode.

The placement of the clusters in the work window does not reflect the anatomic coordinates of the clusters. It is just to allow the user to get a smart organization of the clusters in the work window.

In the automatic mode, the cluster are placed on a circle in the order of their creation. In the manual mode, clusters can be manually positioned using the mouse.

5.10 Output files

Input: `*.lnk`

Output: `*.len, *.wgt, *.clu, *.anat, *.lnk`

5.11 Known problems

No known problems.

Chapter 6

The full featured network editor

The name of the full featured network editor is `network_editor`.

XNBC has a simple network builder, the simple network Link Editor provides the basic features to create the files needed by the simulation:

- Nucleus creation
- Generic intra- and inter-clusters links creation
- Neuron to neuron connecting ability

The simple network editor provides only connection in excite, inhibit or randomized mode and does not take the cluster repartition into account.

The need of the spatial functionality for the network builder made the simple network editor unable to do spatial organization of the network. Nuclei and clusters are not separated: the cluster is only a group of neurons of the same kind (same neuron file). This is often enough, this why the simple network editor is still distributed within XNBC.

6.1 New concepts bring by the full featured network editor

The new concepts came out with the need of a 3D network representation. The cluster concept met here its limits:it is only a cast of neuron , which no real spatial meaning.

Inspired from the brain geometry, comes now the nucleus concept. A nucleus has a spatial meaning: it is made of neurons that can be from several different clusters, and that are placed together in a same area.

So, the new network builder include several editors , that work a different levels :

- The Network Editor, the main editor, where you can find nucleus, neurons and connections between them.
- The nucleus editor, which works at the nucleus level.
- Other editors , like neuron or generic link editor, and so on.

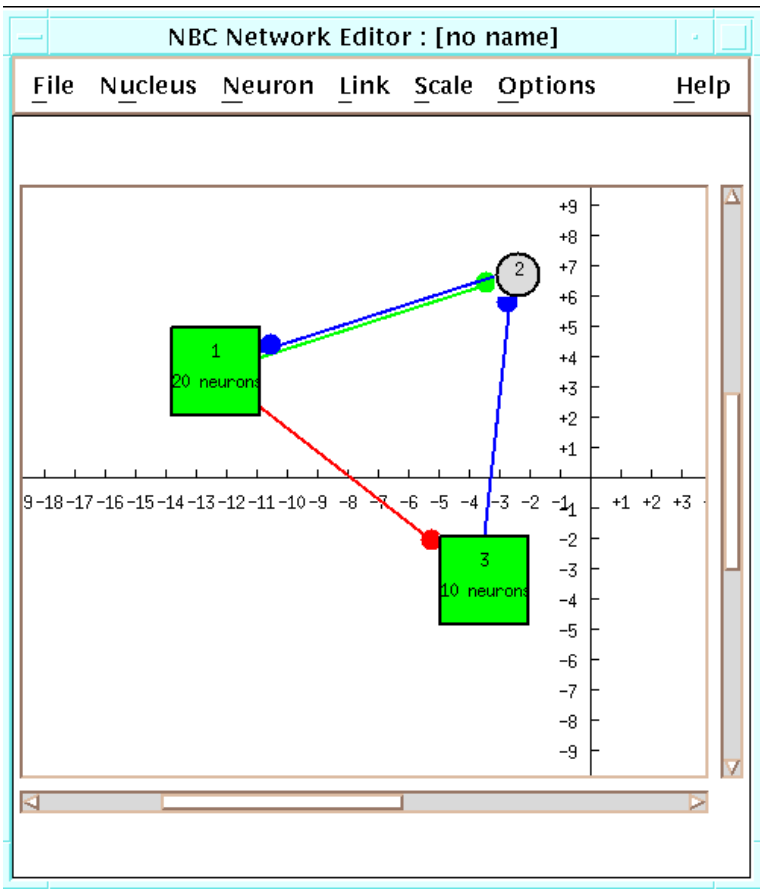


Figure 6.1: **The full featured network editor working space**

6.2 User interface of the full featured Network Editor.

According to the nucleus concept and the spatial 3D representation, the user interface is made of several windows as follows:

- The main window called the **Network Editor** including the network 3D view area and all its relative functions (Fig. 6.1)
- The nucleus editor window called **Nucleus Editor** including the nucleus 3D view area and all its relative functions (Fig. 6.2)
- The nucleus parameters dialog box called **Nucleus Parameters** providing a dialog that can modify nucleus parameters (Fig. 6.3)
- The nucleus composition dialog box called **Nucleus Composition** providing a dialog for adding clusters to a nucleus in generic editing mode
- The neuron parameters dialog box called **Neuron Editor** providing a dialog box for modifying the neuron parameters
- The simple neuron to neuron link editor dialog box called **Single Link Editor** that fixes the single neuron to neuron link values such as the axon length for instance

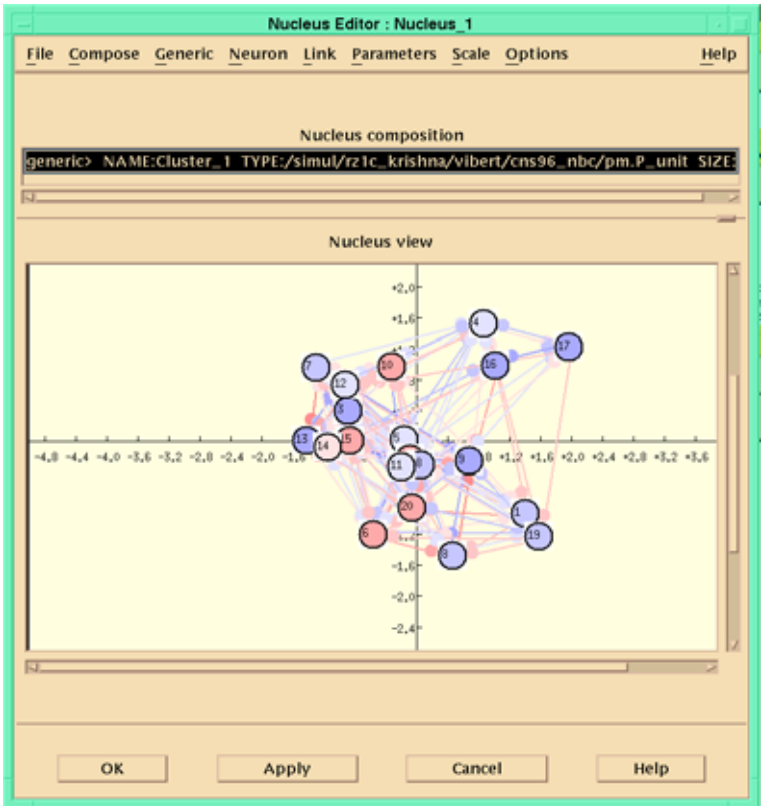


Figure 6.2: **The full featured network editor nucleus editor**

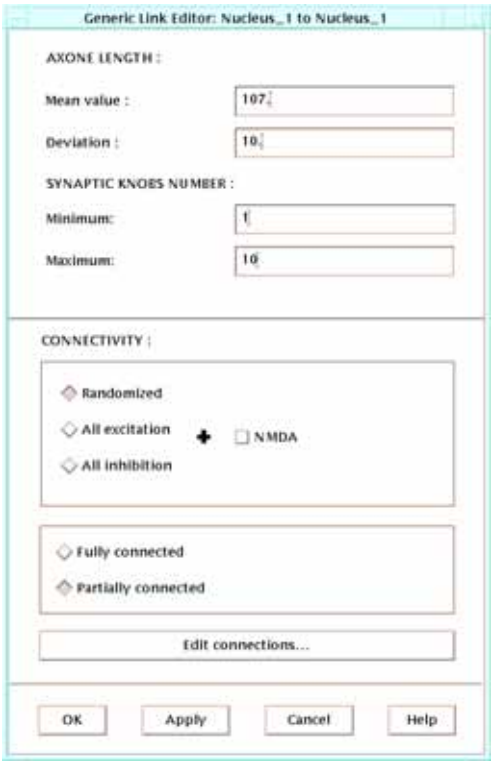


Figure 6.3: **The full featured network Nucleus Parameters dialog box**

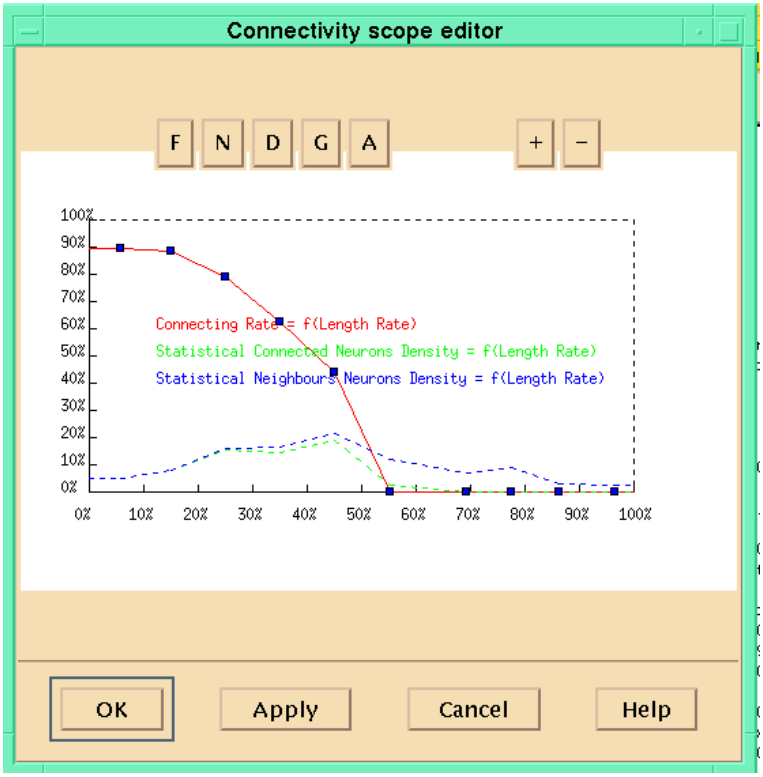


Figure 6.4: **The full featured network editor connection curve editor**

- The adjusting connectivity curve editor called **Connectivity Curve** providing in case of partial generic connection a way to weight the connection repartition according to the nucleus anatomy (Fig. 6.4).
- The generic link and connection matrix editor window called **Connection Matrix Editor** including the 2D matrix view of one particular generic link (Fig. 6.5)

The 3D views are in fact 3 2D views figuring a 3D Horsley Clarke coordinates system. A pop-up menu allows the user to switch between the different L-H, L-AP and AP-H views (the Horsley Clarke coordinates system is inspired from the brain geometry: L for lateral, H for height and AP for antero-

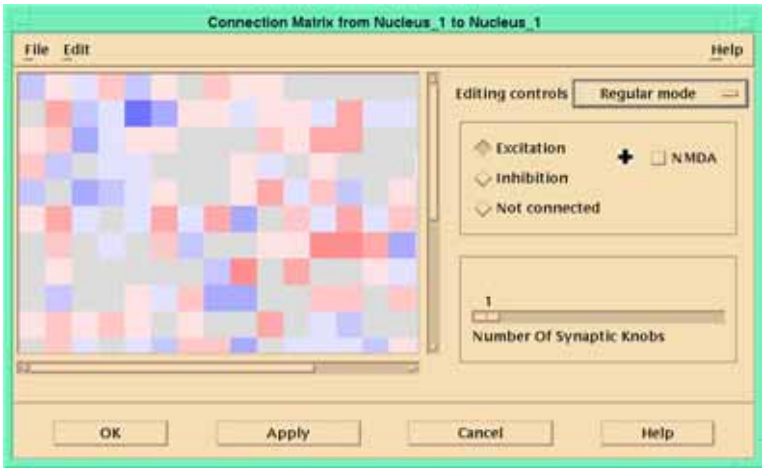


Figure 6.5: **The full featured network editor connection matrix editor**

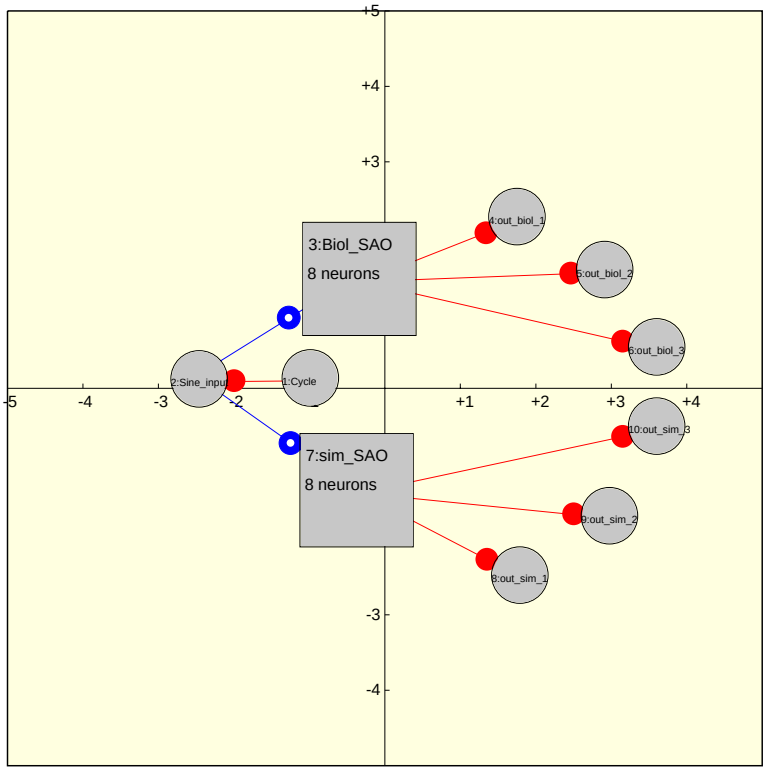


Figure 6.6: A PostScript hard copy of a network

posterior, the origin being in the middle of the axe joining the two ears).The L-H view is called the **Front View**, the L-AP the **Horizontal View** and the AP-H is called the **Sagittal View**.

In addition to that, there are two other functionalities implemented: the ability of redefining the origin in the work area (note that all distance are expressed in millimeters) and also making zooms in or zooms out in each view.

The Look and Feel of the full featured Network Editor follows the concepts of the Motif Style Guide, giving in that the warranty that the user can not loose the control of his work as he already worked with other Motif designed applications. The object orientation is also respected: the user selects the neuron or whatever he wants to select before to invoke any actions on the selected item.

The **Help on line** is implemented at all the levels of the tool certifying that the user always know which action to invoke or the steps to follow before invoking actions. A **Help on context** is also implemented, providing a dynamic help library to allow new topics to be included, such as neuron types documentation or specific biological topics.

PostScript representation of networks are possible through the File Menu/Print (Fig. 6.6).

6.3 Links between neurons and between nuclei

Links (axons) can be established either on a global basis, or individually.

On the network window, links are indicated by a colored line (red: excitatory, blue: inhibitory and green: random -a mix of both excitatory and inhibitory connections). The color of the nucleus indicates the type of intra nucleus connection. A single neuron with a color indicates a recurrent connection of the color type.

On the nucleus window, each individual interneuron links are indicated with the same color code. The color intensity indicates the relative weight (between max and min weight in the nucleus).

Each neuron can be moved to be positioned at the right anatomical position (in the 3D space).

6.4 Network concept

A network is made of one or several nuclei and one or several isolated neurons.

A nucleus is made of several neurons. These neurons can pertain to different clusters.

A neuron is a neural entity that has a particular physical type, and has a specified location.

It is of course a real concept. The neurons are the basis of the neural activities.

Four different models are implemented:

- the Conductance Based Model of neuron (CBM), a Hodgkin-Huxley like model with 14 different transmembranar currents
- the Phenomenologic Model of neuron (PUM), a phenomenologic model with adaptation and post spike membrane shunt.
- the Leaky Integrator Model of neuron (LIM), classical.
- the Burster Unit Model of neuron (BUM), a phenomenologic model of conditional burster with adaptation and post spike membrane shunt.
- the Virtual (not simulated by the simulator) model stored in a file and coming from either a live experiment or a previous simulation. This model allows to made hybrid networks made of simulated neurons receiving inputs from neurons experimentally recorded.

6.5 Nucleus concept

The nucleus is a new abstract concept introduced with XNBC V8.0.

It is a convenience object to design a group of neurons (belonging to clusters) that have the same location area, specified by a center and a radius around this center. This concept introduces the spatial influence in the networks interactions and allows to take into account

- the anatomic location of neurons (the Horsley Clarke coordinates can be used)
- the connection according to the inter neuron distance
- the connection pattern
- the dissociation of anatomical location and unit characteristics
- the neuromodulator or drug concentration according to the production or injection locus (in a future version of XNBC)

6.6 Output files

Input: `*.lnk, .net_customrc`

Output: `*.len, *.wgt, *.clu, *.anat, *.lnk, *.ps, .net_customrc`

6.7 Known problems

There is no known problems, but this tool is probably the most complex, and can appear as little bit complicated to use.

Chapter 7

The drug editor

The name of the drug editor itself is `drug_file_editor`. It allows to create new drug names (normally an existing drug, but it is not necessary) with the transmembranar currents this new drug inactivates. Drugs and currents are saved in a file whose user can modify. A short default list is in a Default.drugs file, not modifiable by users. The user interface is shown on Fig. 7.1. Two lists appears on the panel. One for the drugs, the other for the existing currents.

7.1 Menu bar

A menu bar allows to chose, modify or create a drug file, and to get help by browsing the manual.

File [New] Creates a new file. It displays a file selection box to chose a new name.

[Open to modify] Allows to read and modify an existing drug file. It displays a file selection box to chose an existing file name.

[Open to visualize] Allows to read an existing drug file. It displays a file selection box to chose an existing file name. The file cannot be modified. It is secure option!

[Save as...] Allows to save the data with a new name. The old file is not modified.

[Save] Allows to save the data with the same name. Previous data are lost.

[Exit] Quits the drug editor.

Browse manual Self explaining.

To add a drug, a file to modify or a new file muas have been selected. Enter the new drug name in the text field at left of the panel. Then, press the [Add] pushbutton. The drug name appears in the list. Select it with the mouse (one click selects, one other click remove the drug). Then select the currents the selected drug inactivates in the right list (one click selects, one other click deselects). Several currents can be selected.

When a drug of the list is not selected, clicking on it select it and shows the currents it inactivates by selecting it. It the file was open as to allow modification, it is possible to modify the currents the drug inactivates.

Double clicking on a drug name removes the drug from the list.

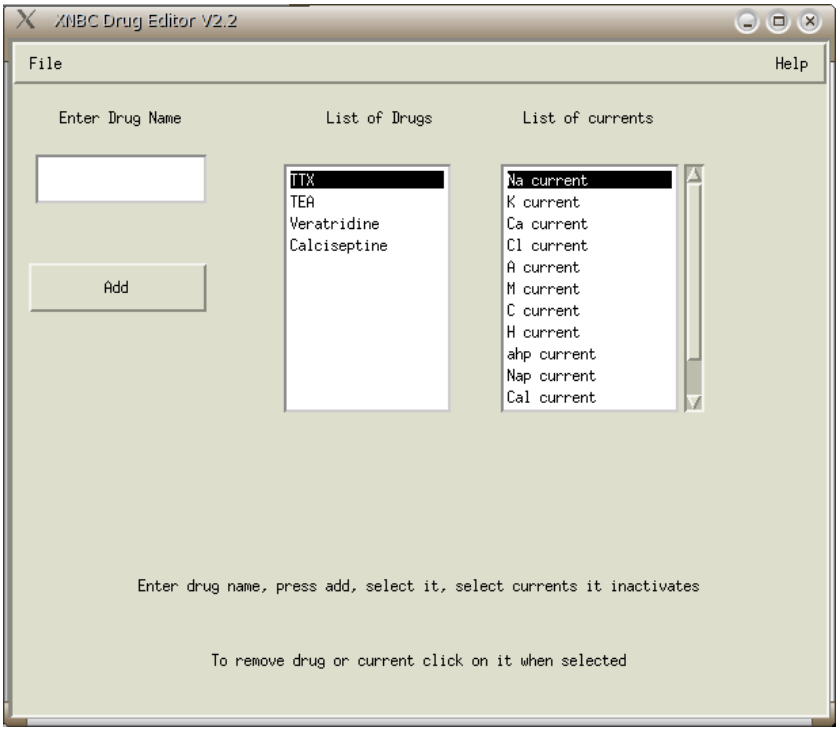


Figure 7.1: The drug editor user interface

Double clicking on a current name deselects the current in the list for the selected drug.

If modifications were not saved, a confirmation is asked on exit.

7.2 Files

Input: `Default.drug, *.drug`

Output: `*.drug`

7.3 Known problems

There is no known problems, but this tool is not very secure, since it is possible to remove inadvertently a drug y a double click. This should be modified in the next release.

Chapter 8

The simulator

The name of the simulator itself is `nbc_x`. XNBC V8 is a software to simulate and analyze biological neural networks models. Normally, the control panel displayed by the xnbc command allow to control all the modeling process. Nevertheless, the simulator itself allows to control also, in the same way, all the simulation process, using pulldown menus instead of graphic pushbuttons (Fig. 8.1).

Two models of neurons are available, an enhanced leaky integrator and a Hodgkin-Huxley type model with 14 different ionic currents.

Inputs to the simulated neurons can be provided by data stored in files from actual experiments, allowing to make “hybrid” networks. The modeled neurons as well as the network are described using graphic tools.

Neuron and network parameters can be modified during the simulation, to mimic electrical stimulations and drugs action.

After simulation the temporal evolution of the network or those of selected neurons can be visualized, and point process, frequency and dynamic analyzes can be performed.

There exist two graphic editors to adjust the neuron parameters.

8.1 The leaky integrator model graphic editor

XNBC’s LIM is derived from the classical leaky integrator implementing properties such as fatigue and post-spike membrane shunt (for a complete description of this model, see Vibert *et al.*, 1994).

The user adjusts the parameters by sliding the scale cursors (right part of the screen), while the temporal evolution of the membrane potential (left part of the screen) changes in real time according to the parameter values. In the LIM, a pacemaker behavior is obtained with a threshold below the resting potential. A small amount of noise can be added to the membrane potential. When the parameters are correctly tuned, the user saves the unit parameters or plot the graph copy.

A conditional burster is also available from the same neuron editor, leading to an BUM (Burster Unit Model).

8.2 The conductance based model graphic editor.

XNBC's conductance model is derived from the Hodgkin-Huxley (HH) model, and incorporates 14 different transmembranar currents. This model explicitly takes into account the Na+, K+, Ca++, and Mg++ ions. The following currents are implemented by XNBC: I_{Na}, $I_{Na+\_persistant}$, I_{Ca}, I_t, I_K, I_M, I_A, I_{AHP}, I_C, I_H, I_{NMDA}, I_{leak}, I_{syn_EPSP} and I_{syn_IPSP}. The currents are modeled using the dynamics of their activation/inactivation constants. All parameters of the CBM can be individually adjusted. The NMDA receptor for glutamatergic synapses neuromodulation is also implemented in this model (for a complete description of this model, see Vibert et al., 1994). The user adjusts the parameters by moving the double dials cursors (or by typing the value), while the temporal evolution of the membrane potential and ionic current (left part of the screen) change in real time according to the parameter values. The graphic submenu allows to plot any parameter versus any other parameter in order to study the dynamics of neuron behavior. Current and voltage clamp experiments can be simulated in order to adjust the conductance values. Drugs such as TTX and TEA can be released to compare the neuron behavior with and without the corresponding blocked channels.

In this CBM the pacemaker behavior is obtained by correctly adjusting the Calcium current and the the I_A current. The small amount of noise added to the membrane potential irregularizes the interspike intervals. When the parameters are correctly tuned, the user saves the unit parameters.

The postsynaptic potential (PSP) parameters, namely their amplitude, rise and decay time constant, depend on the membrane on which the postsynaptic receptor is, and are consequently a neuron characteristic. The three parameters can be adjusted separately, and are modeled as a modification of the membrane potential or a synaptic current I_{syn}, according to the chosen model.

8.3 Assembling the modeled network

The connection matrix of the modeled network can be described using one of the two graphic editors.

The units of one nucleus can be connected with the units of any other nucleus (including itself) through either all excitatory, all inhibitory or both excitatory and inhibitory synapses.

8.3.1 The simple editor

It allows to build nuclei containing only one cluster each. It is made for rapidly build large networks made of several clusters that can be mixed together (CBM, LIM or virtual). It allows to finely edit the connection matrix.

It does not allow to see individually the neurons inside the clusters, nor to work in Horsley-Clarke coordinates. It does not allow to describe the NMDA connections.

8.3.2 The full featured editor

It allows to build nuclei containing several clusters each. It is made for build small (or large if you have time...) networks made of several clusters that can be mixed together (CBM, LIM or virtual). It allows

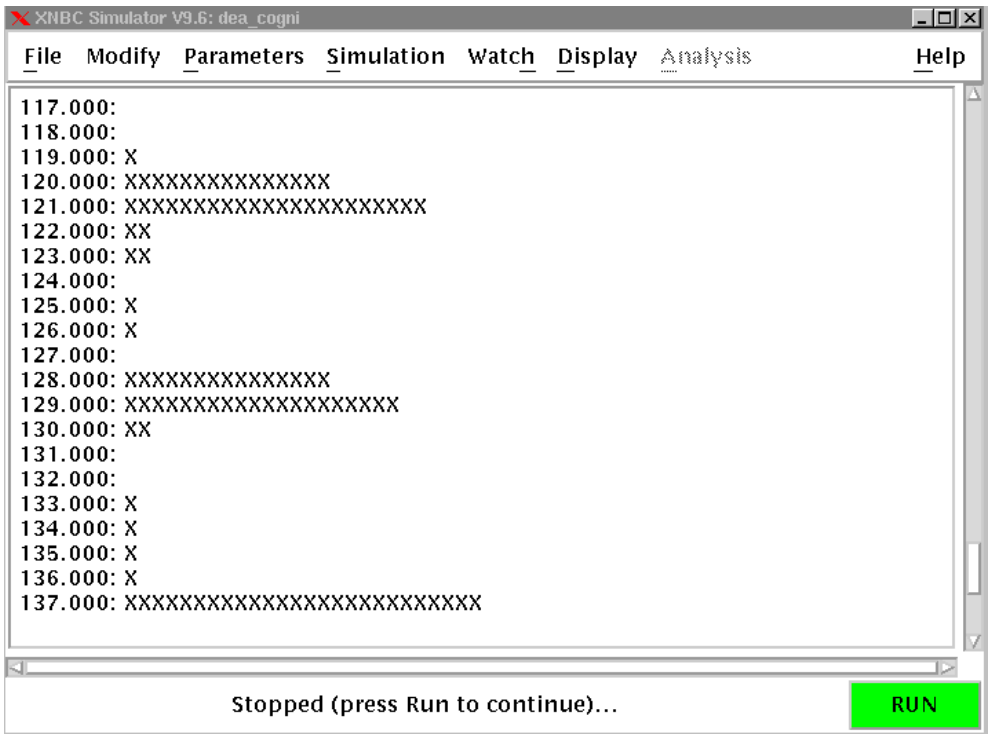


Figure 8.1: **The user interface of the simulator**

to finely edit the connection matrix and to choose the connection density around a given neuron. This is a rather sophisticated tool.

It allows to see individually the neurons and their connections inside the clusters, and to work in Horsley-Clarke coordinates. Connections with glutamate release acting on NMDA receptors can also be specified.

8.3.3 Relationship between nuclei and neurons.

Each neuron, whatever the model chosen, acts on others via an axon whose length, diameter, transmitter release, etc., are reflected by the delay between the spike in the emitting neuron and the postsynaptic potential (PSP) in the receiving neuron. PSPs can be either excitatory (EPSP) or inhibitory (IPSP). The weight of the connections (the number of synaptic buttons) is distributed using an uniform random distribution. The inter- neural transmission delay can be adjusted. The inter-neural delay in a given nucleus is computed from the user defined mean and standard deviation, and represents the cumulated effect of both the length of the axons and the synaptic delay. Connections are setup graphically by first placing the nuclei in the Horsley-Clarke coordinates, then by populating the nucleus with clusters of neurons and describing the intra nucleus connection pattern. Several convenience tools allow to describe precisely these connections (inter neural connection of neighbor neurons according to a probability curve user shaped, crystal editor for repetitive connection patterns, point to point connection editor). Inter nuclei connections are then described using the same tools.

8.4 Loading the network

It is possible to prepare or modify, from the simulator itself, the neuron files or the network files.

This is done using the Menu Modify.

- Modify/PUM neuron: calls back the Phenomenologic model editor (P_unit neurons)
- Modify/LIM neuron: calls back the Phenomenologic model editor (L_unit neurons)
- Modify/BUM neuron: calls back the Phenomenologic model editor for conditional bursters (B_unit neurons)
- Modify/CBM neuron: calls back the Conductance based model editor (G_unit neurons)
- Modify/Simple network: call back the simple network editor
- Modify/Detailed network: call back the full featured network editor

Once the network is built and the iteration step and the duration of the simulation are set, the simulation can be run.

First load a link file (Menu File/New simulation)

Then answer the questions asked by the program (normally answer OK to all the dialog boxes...)

8.5 Running the simulation

Once the network is loaded, the simulation can begin.

Chose the time step (defaulting to 1ms) (Menu Simulation/Time step), and the time of the periodic stop that can be programmed (Menu Simulation/Next stop), and the simulation duration (Menu Simulation/Last stop) -this is optional, since you can stop the simulation when you want, interactively-.

Chose the way you survey the simulation process, either by watching at the individual spikes, or at the global activity of the network (partitioned into nuclei) using the Menu Watch/Dot display (individual spikes) or Watch/Global act... (global activity) or again only a counter (Watch/iteration counter or Watch/milliseconds counter).

The menu Display allows to obtain the connection matrix of weights, or of axon lengths, or again the NMDA connection matrix.

During the simulation process, the network behavior can be observed on a graphic display representation.

At any time the simulation can be momentarily stopped by pressed the red STOP button or by typing (CTRL-I) -(CTRL-R or pressing the green RUN button starts again)- in order to modify the external input feeding one or several nuclei, to give stimulation, drug, or change any parameter. It is also possible to modify some anatomical characteristics, such as a connections between two nuclei, mimicking a lesion, or the membrane properties, mimicking pharmacological effects of drugs (Menu Modify and Menu Parameters). The simulation can be stopped at any time, and the network state kept for a further simulation continuation.

8.6 External inputs

Each nucleus can receive external inputs (this is defined during the simulation, not at the level of the network editor).

It is possible to modify the input or several parameters during the simulation.

This is done using the Menu Parameters.

- Parameters/Nucleus: Allows to modify parameters of the nuclei
- Parameters/Clusters: Allows to modify parameters of the clusters

Then select the nucleus or cluster that must be affected. A dialog box opens, The same dialog box allows to modify all parameters, they grouped into several panels arranged in a ring. Thus select Next to get the following set of parameters. The OK pushbutton closes the dialog box and validates the modified parameters.

Background Gaussian noise, with adjustable mean activity and variability representing the activity of neurons external to the simulated network of neural clusters can be added for each nucleus. Neurobiological experiments, such as electrical stimulations, can also be simulated. XNBC allows to include virtual clusters, treated exactly as the other clusters regarding the connections with the "true" (i.e. simulated) clusters, but where neurons are not modeled. The file may come from a previous simulation or actual experimental data.

8.7 Stopping the simulation

At any time the simulation can be momentarily stopped by stroking CTRL-I (Tab) or the red STOP button or by typing (CTRL-I) -(CTRL-R or pressing the green RUN button starts again)- in order to modify the external input feeding one or several nuclei, to give stimulation, drug, or change any parameter. It is also possible to modify some anatomical characteristics, such as a connections between two nuclei, mimicking a lesion, or the membrane properties, mimicking pharmacological effects of drugs.

The simulation can be started again from the same state by stroking CTRL-R or pressing the green RUN button. The simulation can be stopped at any time [CTRL-I or (Tab)], and the network state kept for a further simulation continuation.

Before to be able to access the simulated data, it is necessary to close the simulation (Menu File/Close (and save files)). This enables the Menu Analysis.

8.8 Analysis tools

When the simulation is closed, data files can be either visualized or analyzed. Analysis can be made at the level of the individual spike trains using time series data analysis tools, or at the level of the global activity of a nucleus.

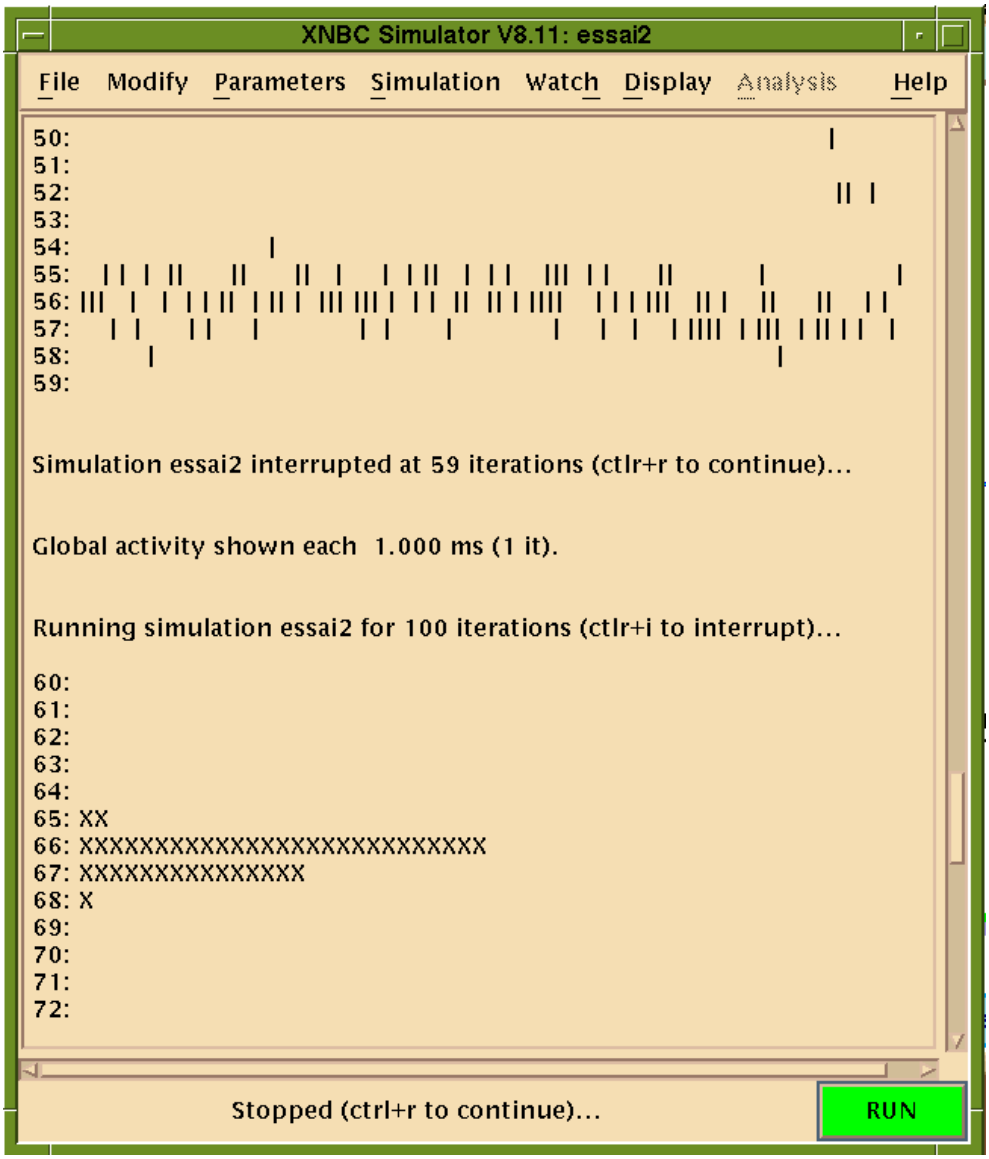


Figure 8.2: The two representations available during the simulation: dot display and global activity

8.8.1 Visualizing the simulation result

After the simulation, the color visualization tool can display the behavior of the modeled network along time, using several representations with an adjustable speed and scale. This visualization tool can be seen as a video tape player allowing to run, stop, go to a given iteration, or change the playing speed. Several representations can be displayed (Fig. 8.2:

- the joint color coded temporal evolution of the membrane potential of simulated units (and of units from a virtual cluster).
- the intracellular recordings of some of the simulated units, using the CBM or the LIM.
- the global activity of each nucleus (percentage of active units by ms)
- the spikes traveling along axons. This representation is convenient to study the synchronization of spike trains in the network.

8.8.2 Time series analysis

One is for time series analysis, and is mainly devoted to analyze the individual unit discharges (point process analysis: rate time, interspike intervals, pre and post-stimulus histograms, Poincaré maps of intervals, auto and cross correlograms, etc.). It offers 31 different analysis grouped into 8 submenus. For example it can show interspike interval evolution along time for the noisy sine input and the corresponding cycle triggered histogram (CTH) , a post event histogram triggered by a cycle start.

8.8.3 Cluster activity analysis

The other analysis tool is devoted to the analysis of the clusters activity and allows 8 different analysis with FFT, Poincaré maps, auto and cross amplitude correlograms, etc.

8.9 Help

It allows to browse the manual.

8.10 Output files

Input: `*.lnk, *.def, *.len, *.wgt, *.clu, *.anat, *.tms, *.neur`

Output: `*.tms, *.mod, *.sim, *.def, *.pmb, *.prm, *.sim, *.neur`

8.11 Known problems

Starting from a saved simulation to continue, existing from version 3, is not yet re-implemented in version 8.

Chapter 9

The visualization tool

The name of the visualization tool is `visu_nbc`. Visu_nbc is a tool to visualize biologicals neural networks activity. This tool is also accessible from the xnbc menu : Applications->Visualization

To visualize the network connectivity, visu_nbc needs the files describing the connectivity matrix (*.len and *.wgt). To visualize the evolution of the network activity, visu_n_b_c needs a a *.def* (simulation definition) and a *.pmb* file (membrane potentials).

9.1 Available representations

Four representations are available. They are accessible either using the *Option/Type* of representation menu, or more directly by pressing one of the four pushbuttons in the menu bar. These representations are:

- **Linear Representation**
  This representation is like a dot display but shows as a color graph the evolution of the membrane potential between spikes (Fig. 9.1, left). Each colored square represents a neuron. Each column represents one iteration of the simulation. The black lines delimit clusters.

  With this representation, it is also possible to select one of the neurons with the pointer and to depress the right mouse bouton to have a representation of the intra cellular recording of the selected unit (Fig. 9.1, right). The time scale can be zoomed at will by selecting with the mouse middle bouton 2 limits. The right mouse bouton go back to the original scale.

- **Global Activity Representation**
  This representation displays for each cluster its global activity in percentage at each iteration (number of spiking neurons / number of neurons in the cluster) (Fig. 9.2).

- **Intracellular Activity Representation**
  This representation displays for each selected unit the temporal evolution of the membrane potential (Fig. 9.3).

- **Matrix Representation**
  This representation displays the network state at each iteration. The network is represented like a matrix. Each colored square represents a neuron. Each black big square represents

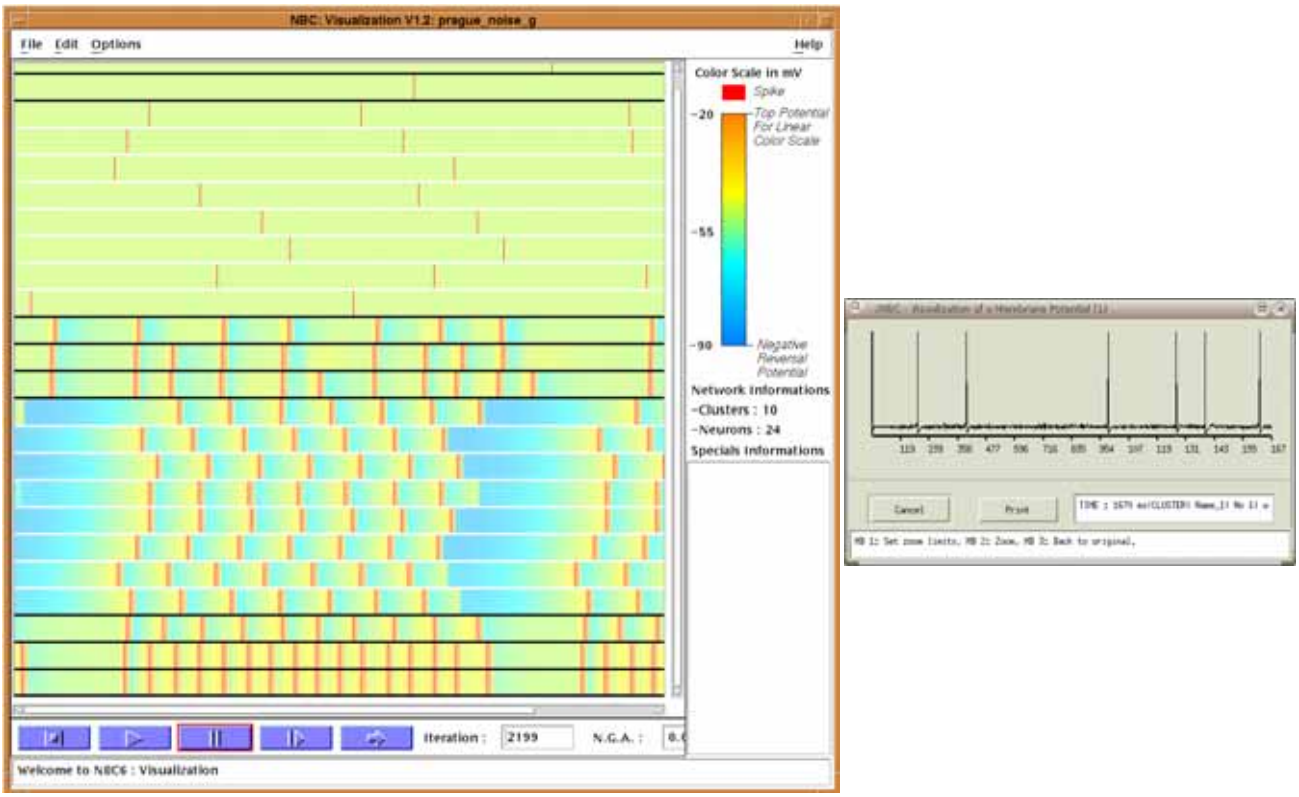


Figure 9.1: Representation of membrane potential neurons with the visualization tool

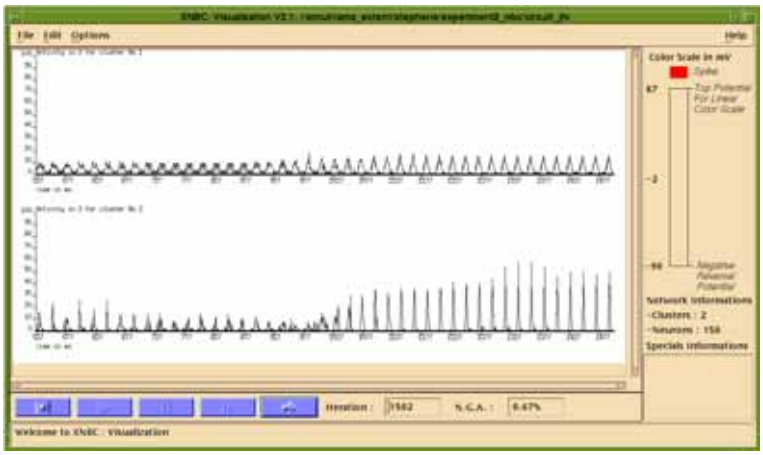


Figure 9.2: Representation of nucleus global activity with the visualization tool

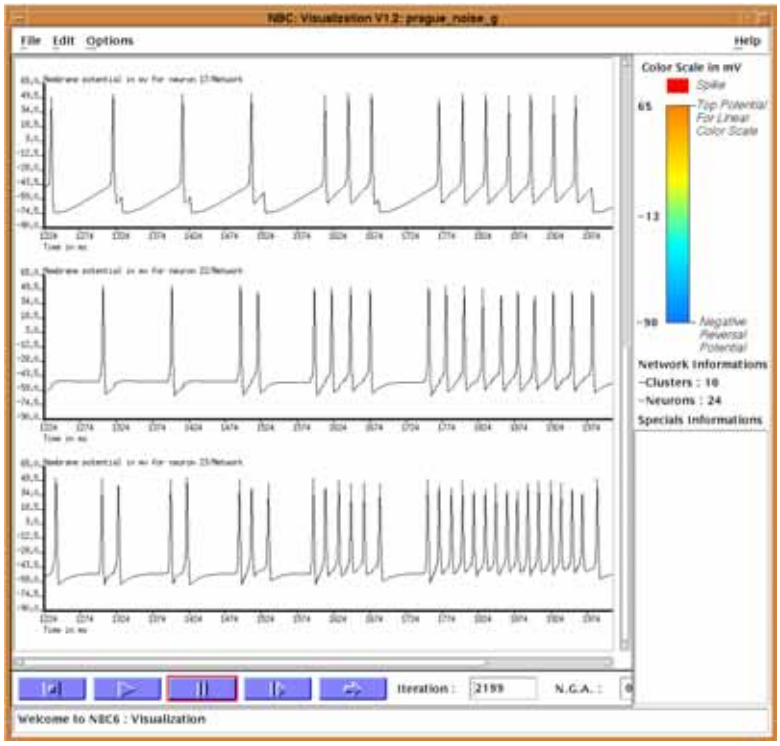


Figure 9.3: Representation of intra cellular recordings with the visualization tool

a cluster. Small lines represent connections from a neuron to another. With **Show Details** connection types can be visualized. When a neuron fires the spike can be seen traveling along its axon toward others neurons (Fig. 9.4).

9.2 The user interface

- **Menu File**

  Open Network Files : choose a *.cnx* file.

  Open Simulation Files : choose a *.def* file.

  Close Simulation : close simulation files.

  Exit : exit visu_n_b_c.

- **Menu Edit**

  Show Details : visualization of details for the selected. representation (Button On/Off).

  Refresh Screen : redraw the selected representation.

  Print Network : print the selected representation.

- **Menu Options**

  Scale Top Value : permits to change the top value of the membrane potentials color scale.

  Time Step : permits to modify the time between the visualization of each step of the simulation.

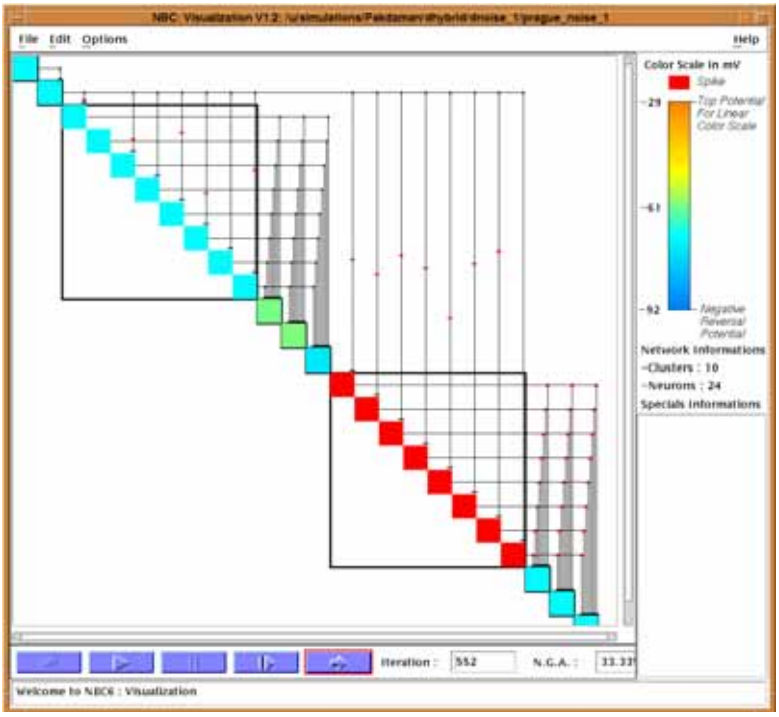


Figure 9.4: Representation of spikes on axons with the visualization tool

Type of representation : cascaded menu allowing to select the type of representation. It is a duplication of the pushbuttons on the menu bar.

Linear: color dot display with the membrane potential represented by a color

Matrix: representation of the neurons on the diagonal with their connections with the other neurons. Spikes are seen traveling on the axons (small red bars).

Global Activity: Activity of clusters (not of nuclei!) represented by the percentage of simultaneously firing units in a given cluster.

Membrane Potential: Active only if at least one neuron was selected. Represents the intracellular recording of selected neurons.

- **Menu Help**

  Browse manual: self explaining.

  About Visualization: General informations.

9.2.1 Color Scale

This scale associates a color at a value of the membrane potential. This scale has two parts :

The first one (red rectangle) indicates the potential for a spike. This value is read in the *.def* file.

The second one indicates a linear correspondence between color and potential values. The top value can be modified (the bottom value is read in the *.def* file as the **Negative Reversal Potential**). By increasing this value, small variations of the potential can be filtered, and on the opposite by decreasing it small variations of the potential around the resting potential can be magnified.

To change this value :

- select the corresponding menu in the Options menu.
- move the cursor of the scale in the dialog box to set the wanted value.
- click on OK to change the value or on Cancel to keep the previous value.

9.2.2 Time Step

The time between representation of each step of the simulation can be modified. This time acts on the visualization speed not on the simulation time step increment. The major purpose of this function is to slow down the visualization when the network is small or the computer too rapid.

To change this time :

- select the corresponding menu in the Options menu.
- move the cursor of the scale in the dialog box to set the wanted time.
- click on OK to change the time or on Cancel to keep the previous time.

9.2.3 Visualization buttons

These buttons are in the bottom left corner of the window. They control the visualization like a **video tape player**.

The first button at left stops the visualization and returns to the first iteration.

The second button starts the visualization.

The third button pauses the visualization at its current iteration. Visualization can be restarted from this iteration by clicking the second button.

The fourth button allows to go directly to a specific iteration. Enter the selected iteration number in the iteration text field, then press *return* or click on this button to go to the selected iteration.

The fifth button runs one step of the visualization (step by step visualization).

9.2.4 Informations fields

Several fields display informations on the window.

`Iteration` : display the current iteration. This field allows too to enter an iteration for the "Goto Index" button (see previous section).

`N.G.A.` : display the Network Global Activity in percentage (number of spiking neurons / number of neurons in the network).

`Network Information` : display number of clusters and neurons in the network.

`Special Informations` : display different informations according to the selected representation. To obtain this information click in the window where the visualization is displayed. Then a hair-cross cursor, moving with pointer device, appears in the window. Move the cursor to point on which part of the visualization informations are needed.

Print Network

To print the visualization at its current iteration (when the visualization is paused) select the corresponding menu in the Edit menu.

Then choose the PostScript printer, if the file must be kept or not after printing, the type of the printed part: entire representation or only the visible part (what is currently seen in the window). Then click on OK to print or on Cancel to cancel it.

If the PostScript file is kept for further printing, a specific file name can be given (default name is *file_name.ps*.

Open Files

To open Network files and Simulation files select the corresponding menu in the file menu. Then select file in the dialog box and click on OK to open it or on Cancel to cancel it.

9.3 How to visualize data using the visualization tool

`visu_nbc` is like a "video tape player".

A normal session is as follow :

- Open Network file (*.cnx*) then the representation of network appears in the window. Default representation is LINEAR REPRESENTATION (see below).

- Open Simulation files (*.def*). visu_n_b_c tries to open the given file name with *.def* and *.pmb* extension. If it fails the visualization is impossible and an information message is sent. A file name without extension, with a *.def* extension (default int the file selection box) or with a *.pmb* extension can be given.

- Now the visualization is ready to start.

- At every moment the representation type can be changed, or special informations on the current simulation can be obtained.

- To print the network, change time step or change color scale top value, the visualization must be paused.

- At the end of the work, close the simulation.

- Others Simulation files for the same network or an other network can be obtained.

9.4 Output files

Input: `*.len, *.wgt, *.lnk`

Output: `*.ps`

9.5 Quick selection and visualization of membrane potential of neurons

Using the second (MB2) and third (MB3) mouse buttons, it is possible to quickly select or obtain a membrane potential recording.

The MB2 allows to directly select neurons on the linear representation. If not already visible, it displays the neuron selection/unselection box. It has the same effect as if going through the menu.

When the file reading is completed, it is possible to directly obtain a small window with the membrane potential of the neuron selected using the MB3. WARNING: the plot begins a time 0 and finishes where you selected the neuron. If the MB3 is depressed in the middle of the panel, you get only from 0 ms to about the half simulation time; thus, selection should be done at the right part of the panel. The choice of a subpart of the trace is possible in the panel using the MB1, MB2 or MB3. MB1 is used to set the limits of the part of interest (a vertical red line is drawn at the chose place, and can be displaced while the mouse button is pressed). Red lines can be drag using MB1 again. When the limits are OK. MB2 zooms the part of interest (the window is re-sizable at will). Zoomed view can be re-zoomed at will. MB3 goes back to the original.

A button *Print* allows to print the membrane potential.

A button *Cancel* closes the window.

Up to six neurons (windows) can be visualized simultaneously in this way.

9.6 Known problems

No known problems.

Chapter 10

The time series analysis tool

The name of the time series analysis tool is `xtms`. Xtms is a tool devoted to the analysis of time series of point process. XTMS stands for *X Window* TiMe Series analysis

XTMS allows to make time series analysis from files storing events as point process. It can analyze the XNBC cluster behavior from the NBC *.tms files. The default input file extension is .tms, the output file is a PostScript file.

10.1 Data format

File has the extension .tms and is a simple ascii text file that can be produced by any text editor or spreadsheet.

The input file format is:

```
1.0 2 2.2 3 3.1 3 3.2 2 3 1 5.0 1 ...
```

were the first value is the time (in increasing order), that can be a float or an integer. It is supposed to be given in ms. The second is the event number. TMS can process simultaneously up to 5 different events. More can be present in the file, but only 5 can be chosen among them, which is largely enough, since only 1 to 3 are relevant for each processing. Up to 50,000 such pairs can be stored and processed. Only the data corresponding to the chosen events are stored.

10.2 XTMS menus

10.2.1 Main menu

The main menu contains the items :

'File', 'Edit', 'Operations' and 'Options'.

- The 'File' menu is a standard and let open and close a source file, save or print a postscript file. This sub-menu contains the items :

  'Open...', 'Close', 'Save PostScript File', 'Print', 'Print on...'and 'Exit'.

'Print On...' : allow the user to select another printer instead of the default one (deactivated, use the environment variable PRINT_PLOT_PS -see Output Help-).

'Open...' : ask the user to enter a file source name and a postscript file name for its work. The user can change of source file by opening a new file.

- The 'Edit' menu allows the user to add restrictions to the data analysis by acting on filters and on the axis characteristics. This sub-menu contains the items :

  'Filters...' and 'Axis lenght...'

  'Filters...' : display a window where the user can select for which data he wants a filter and each filter characteristics.

  'Axis lenght...': display a window that let the user select the axis length.

- The 'Operations' menu allows the user to select the data and to execute the calculus functions. This sub-menu contains the items 'Parameters...', and the groups of calculus functions.

  'Parameters...': open the parameters window

  The calculus functions are grouped together. For each group correspond a sub-menu (A, B, C,... H)

- The 'Options' menu allows the user to select an automatic spooling (automatic printing) or not and to write an *.isi file or not for special calculi functions, or to use default values or not. Not using default values allow to modify a lot of parameters.

- The 'Help' menu gives an "Browse manual' and an 'Help on topic'. This use a dynamic help library to allow new topics to be included.

10.2.2 The xtms calculi functions

In xtms there are 31 functions. In general a user wants to draw a series of graph and will enjoy to have a menu always visible to access quickly to functions. This is why all functions and parameters menus of the main window are 'Tear-off' menus. That means that the user can display permanently all groups of functions menu on the screen...

The main menu propose some general actions and the following submenus (Fig. 10.1):

```
A f(Time) drawings B f(Rank) drawings C Correlation analysis D
Poincare\'s Maps E Inter and Peri Event Analysis F 3D drawings
G Phase map analysis [dx/dt = f(x)]

And the submenus contain the following analyses:

    SUB-MENU A: f(Time) drawings

A Instantaneous rate = f(time) (1 event) [line drawing] B
Instantaneous rate = f(time) (1 event) [dots drawing] C
```

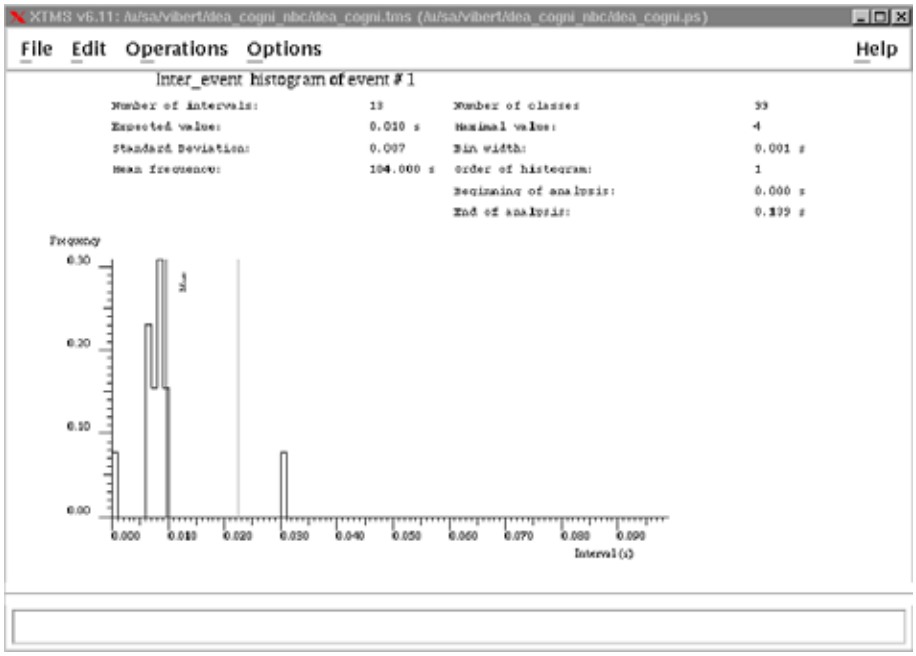


Figure 10.1: **The time series analysis tool interface**

Instantaneous rate = f(time) (2 events) [dots drawing] D Rate as funtion of running time (1 event) [histogram] E Interval = f(time) (1 event) [dots drawing] F Interval = f(time) (2 events) [dots drawing] G Phase = f(time) (2 events) [dots drawing] H Phase = f(interval) (2 events) [dots drawing]

SUB-MENU B: f(Rank) drawings

A interval = f(rank) (1 event) [dots drawing] B phase = f(rank) (2 events) [dots drawing] C power spectrum of rank (1 event) [histogram]

SUB-MENU C: Correlation analysis

A autocorrelation (1 event) [histogram] B crosscorrelation (2 events) [histogram]

SUB-MENU D: Poincare\'s Maps

A Poincare's map of intervals (1 event) [dot drawing] B 2 Poincare's maps of intervals (2 events) [dot drawing] C Poincare's map of phases (2 events) [dot drawing] D Poincare's map of cyclic data (2 or 3 events) [dot drawing]

SUB-MENU E: Inter and Peri Event Analysis

```
A Inter event histogram (1 event) [histogram] B Dot display (2
events) [dot drawing] C Lissajous of cyclic data (2 events)
[dot drawing] D Post event histogram (2 or 3 events)
[histogram] E Post event interv pooled (2 or 3 events) [dot
drawing] F Post event rate pooled (2 or 3 events) [dot
drawing] G Post event phase pooled (3 events) [dot drawing] H
Scatter diagram (3 events) [dot drawing]

    SUB-MENU F: 3D drawings

A Interval I1 = f(I2,I3) (1 event) [dot drawing] B Phase P1 =
f(P2,P3) (2 events) [dot drawing]

    SUB-MENU G: Phase map analysis [dx/dt = f(x)]

A d(interval)/dt = f(Interval) (1 event) [dot drawing] B
exp(d(interval)/dt) = f(Interval) (1 event) [dot drawing] C
d(phase)/dt = f(phase) (2 event) [dot drawing]
```

For each analysis, specific question are asked. All questions admit a default answer indicated in the text fields.

10.3 Changing the parameters

The calculi functions need some parameters to design the drawings. Some parameters are in common or generally used. Those parameters have been implemented in the 'Parameters' window . This window is used to select and display the data to study and its characteristics.

The axis length is accessible in the 'Axis options' window.

The next part of the interface is functions oriented is not the same in xtms and xcaa. For each functions the parameters that are not used are grayed and insensitive in the next windows .

10.3.1 Windows.

Each groups are organized per functions type. the selection of any function in the 'Operations' menu display the same window which name is the selected function's name. This window display and allows to chose the data for the actual function. It also allows to select some parameters necessary for organization of the data on the graph. The parameters unused are grayed...

Some data are always preset to make quicker the operation. But some parameters need the data identification and other parameters (from the window) to display default value.

The 'Compute' button make a compute operation and fill with default value all fields that need to. Also on this window, a 'Compute' button executes the same action than 'OK' button but close the window and display other windows for more specifics parameters or/and compute the drawing. After computing, the drawing is displayed.

There are 4 others windows for specifics functions that need more parameters. The 'Poincaré's map options', 'Arrows options', 'Histogram options' and the 'Three D options' windows allow the to chose default or particular value for the sensitive fields.

10.4 Printing

The 'Options' menu allows the user to select an automatic spooling (automatic printing).

Results of XTMS are given as PostScript graphics, displayed on the X window, and/or automatically spooled (or if asked from the menu area).

The PRINT_PLOT_PS environment variable should be positioned with the right print command and printer indicated.

For example in csh:

```
setenv PRINT_PLOT_PS "lpr -Plaser "
```

10.5 Output Files

Input: `file_in.tms .tmsrc`

Output: `file_in.ps .tmsrc`

10.6 Known problems

No known problems

Chapter 11

The network activity analysis tool

The name of the network activity analysis tool is `xcaa`. Xcaa is a tool devoted to the analysis of cluster activity analysis XCAA stands for Cluster Activity Analysis of XNBC data XCAA (Cluster Activity Analysis) allows to analyze the XNBC output behavior.

11.1 Data format

File has the extension .sim and is a simple ascii text file that can be produced by any text editor or spreadsheet.

The input file format is:

```
"Total Act.  0 0 0 1 0 0 2 0 0 3 0 0 4 0 0 5 0 0 6 0 0 7 0
0 ...  97 0 0 98 0 0 99 0 0 100 0 0 101 1 0 102 2 0 103 5
0 104 1 0 105 1 0 106 0 0 107 0 0 108 0 0 109 1 0 110 0 0
...  3593 0 0 3594 0 0 3595 0 0 3596 0 0 3597 0 0 3598 0 0
3599 0 0 3600 0 0

"1 Nucleus_1 0 0 0 1 0 0 2 0 0 3 0 0 4 0 0 5 0 0 6 0 0 7 0
0
```

The first value is the time, that is an integer. It is supposed to be given in ms.

The second is the number of units that have discharged at this time.

The third is the value of a supplemental input (here null).

The first set of data is the total activity of the network.

The second is the first cluster (or nucleus), etc...

The limit between the series of data is given by the black line and the line beginning with a double quote (") followed by the name of the cluster.

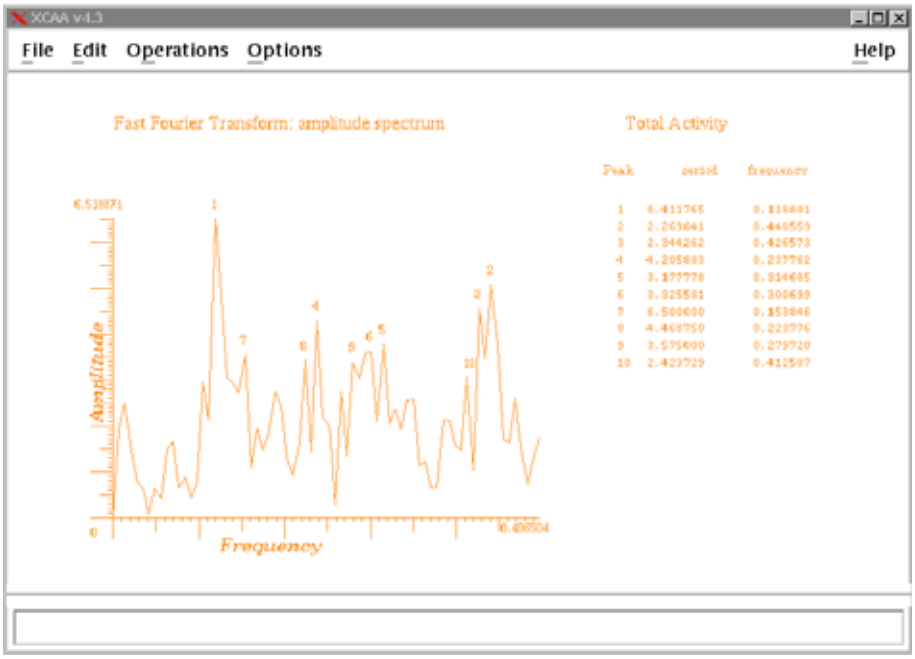


Figure 11.1: **The network activity analysis tool interface**

11.2 XCAA menus

XCAA is X menu driven. XCAA has been implemented with the same main menu as xtms. So the user knows how to use both programs by using one. The main menu contains the items : 'File', 'Edit', 'Operations' and 'Options' (Fig. 11.1).

- The 'File' menu is a standard and let open and close a source file, save or print a postscript file. This sub-menu contains the items : 'Open...', 'Close', 'Save PostScript File','Print' , 'Print on... 'and 'Exit'.

  'Print On...' : allow the user to select another printer instead of the default one.

  'Open...' : ask the user to enter a file source name and a postscript file name for its work. The user can change of source file by opening a new file.

- The 'Edit' menu allows the user to add restrictions to the data analysis by acting on filters and on the axis characteristics. This sub-menu contains the items : 'Filters...' and 'Axis lenght...'

  'Filters...' : display a window where the user can select for which data he wants a filter and each filter characteristics.

  'Axis lenght...': display a window that let the user select the axis length...

- The 'Operations' menu allows the user to select the data and to execute the calculi functions. This sub-menu contains the items 'Parameters...', and the groups of calculus functions.

  'Parameters...': open the parameters window The calculus functions groups correspond to the list of calculus groups in the old version main menu (from the 4th item to last one). For each group corresponds a sub-menu as for the old version (menu A, B, C,... H) excepted the 'Quit' item.

As there are only 8 functions a simple menu is used to call all functions. There is no groups of functions.

The ‘Operation’ menu propose some general actions and the following submenus:

```
A Fast Fourier Transform: amplitude spectrum B Fast Fourier
Transform: phase spectrum C State space of a cluster.  D State at
t, (t+1) (Poincare map) E Amplitude as a function of time F
Amplitude autocorrelation function G Amplitude cross-correlation
function
```

- The ’Options’ menu allows the user to select an automatic spooling (automatic printing) or not and to write an *.isi file or not for special calculus functions.
- The ’Help’ menu gives an "Browse manual’ and an ’Help on topic’. This use a dynamic help library to allow new topics to be included.

11.3 Changing the parameters

The calculus functions need some parameters to design the drawings. Some parameters are in common or generally used. Those parameters have been implemented in the ’Parameters’ window (Fig. 11.2). This window is used to select and display the data to study and its characteristics (asked before the first menu in the old versions).

The axis length is accessible in the ’Axis options’ window.

The next part of the interface is functions oriented is not the same in xtms and xcaa. For each functions the parameters that are not used are grayed and insensitive in the next windows .

For each analysis, specific question are asked. All questions admit a default answer indicated at the prompt.

As for xtms, there is a window called by the selection of any function. This window takes the current function’s name and is organized in 3 parts.

The first part identifies the only data selected and is not editable. The second part is only used for 3D graph parameters and the last part concerns the correlations parameters. As for xtms an ’Apply’ button compute the default values and the ’Compute’ button do the same thing and create the graph.

For each analysis, specific question are asked. All questions admit a default answer indicated at the prompt.

11.3.1 Windows

As for xtms, there is a window called by the selection of any function. This window takes the current function’s name and is organized in 3 parts.

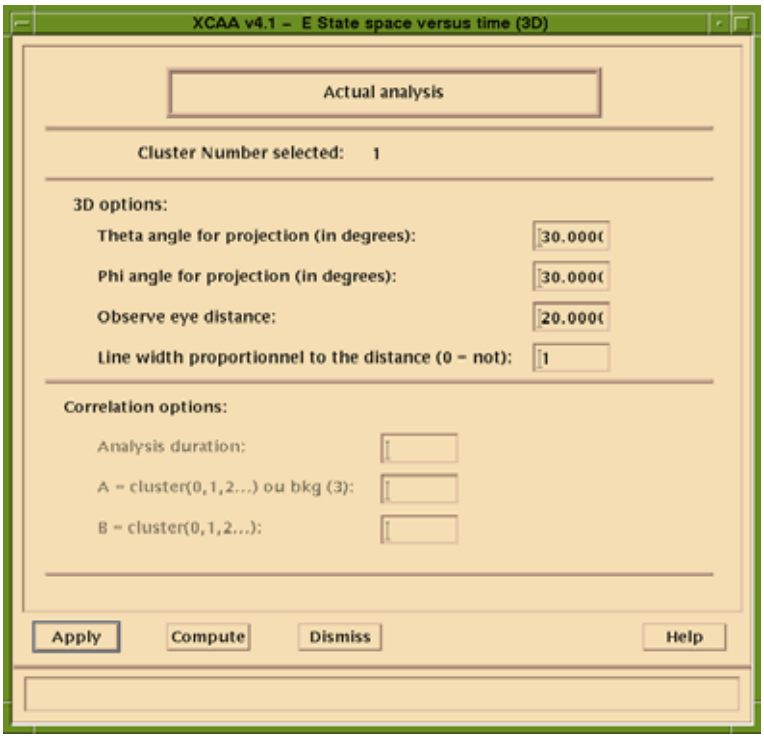


Figure 11.2: The network activity analysis tool parameter pane

The first part identifies the only data selected and is not editable. The second part is only used for 3D graph parameters and the last part concerns the correlations parameters.

As for xtms an 'Apply' button compute the default values and the 'Compute' button do the same thing and create the graph.

11.3.2 Printing

Results are given as PostScript graphics, displayed on the X window, and/or automatically spooled (or if asked from the menu area).

The 'Options' menu allows the user to select an automatic spooling (automatic printing).

Results of XCAA are given as PostScript graphics, displayed on the X window, and/or automatically spooled (or if asked from the menu area).

The PRINT_PLOT_PS environment variable should be positioned with the right print command and printer indicated.

For example in csh:

```
setenv PRINT_PLOT_PS "lpr -Plaser "
```

11.4 Output files

Input: `file_in.tms .caarc`

Output: `file_in.ps .caarc`

11.5 Known problems

No known problems

Chapter 12

Installation of XNBC

Installation of XNBC is based on Imake generated Makefiles. Consequently, you must have installed the xmkmf and imake programs, as well as the Imake template files. This is generally distributed with the X11 distribution, and often installed by default (but not on AIX were you have to build it by following the /usr/lpp/X11/Xamples/README). HPux installation is also done using a special handwritten file, since the template files for imake are not distributed with HPux.

We succeed to install XNBC V8 on

- Ultrix,
- Digital Unix (OSF1),
- Sun OS,
- Linux,
- AIX,
- HPux.

12.1 Customization

If you build XNBC on a Unix system, you should run the `install_xnbc8` script:

```
host> sh install__xnbc8
```

On a Linux system you must have installed Moo-Tif (Lesstiff, the freeware version of Motif is not yet completed, all programs compile correctly, but the menu button don't work, thus...). Since the XFree86 Imake templates do not define the Motif libraries they must be indicated namely. To be able to compile the XNBC package, you must have installed the C development package (Gnu gcc) and the Linux sources (system include files are necessary). Then you have to build XNBC by running the `xnbc.install.linux_version` script (or download a prebuild image -see below-).

Questions about your system specificities are asked. Default answers are provided.

On Linux, you must know were your XFree86 distribution is installed. Since some systems have Xlib in `/usr/X11R6` (redhat) and other on `/usr/X11` (a link) two separate files are provided. Nevertheless, Linux users can download a precompiled version of XNBC8 statically linked with the Moo-Tif libraries in order keep XNBC free from any commercial associated product.

The process takes from 3 mn on a Pentium II/333, 6 min on an AlphaStation 600/266, 15 mn on a Pentium 90 Mhz, up to 45 mn on a DecStation 5000/240.

At the end you have two files `xnbc_env_system_wide` and `xnbc_env_private` containing the environment variable necessary for XNBC, according to your installation in order to define the environment variables.

The `xnbc_env_system_wide` must be used if XNBC is installed in the `/usr/local` filesystem. The `xnbc_env_private` must be used if XNBC is left in the installation directory.

You must verify in both file that the `XNBC_BIN_REF_XTERM` points on the correct Xterminal emulator, that the `XNBC_BROWSER` points on the correct web browser, and that `XNBC_BIN_REF_PSVIEWER` points in the correct PostScript viewer.

Nevertheless, for your convenience, the system wide file is copied locally as `xnbc` and defines the needed environment variables. This provides the simplest way to adapt XNBC8 it to your local installation. Call `xbnc` (the script) that will call `xnbc8` (the actual xnbc program).

If you need to use the programs individually (uncomment the lines flagged for this in the installation script), you can remove the call to `xnbc8` at the end of the `xnbc_env_system_wide` (or the `xnbc_env_private`) script and the source it in order to define the environment variables. For example, xtms can be used to analyze actual experimental data and be useful without the simulator.

12.2 Local Install (user mode)

This mode is automatically selected if you are not superuser when you build xnbc. In this mode the system is installed in the XNBC8 tree :

- `"TOP"/XNBC8/bin` ⇒ binaries
- `"TOP"/XNBC8/uid` ⇒ uid files for Motif
- `"TOP"/XNBC8/lib` ⇒ object libraries

For this version the man directory stays empty after the installation, but all the documentation can be found in `"TOP"/XNBC/doc`.

12.3 Global Install (root mode)

You MUST be in superuser mode. In this mode the system is installed in the XNBC8 tree **and** in the system tree.

- `"TOP"/XNBC8/bin` ⇒ binaries
- `"TOP"/XNBC8/uid` ⇒ uid files for Motif

- "TOP"/XNBC8/lib ⇒ object libraries
- /usr/local/bin/nbc.bin/V8 ⇒ binaries
- /usr/local/bin/nbc.bin/V8/xnbc8_manual ⇒ html manual
- /usr/local/lib/uid ⇒ uid files for Motif Installation

Normally,

- binaries are expected to be in the /usr/local/bin/nbc.bin/V8 directory
- uid in the /usr/local/lib/uid directory.

If the binaries are stored in another directory, the environment variable XNBC_BIN_DIR_TOOLS must be set to this directory. uid files must be stored /usr/local/lib/uid or in the current directory, or in another directory named in the environment variable XNBC_UID_DIR.

If the binaries are stored in another directory, the environment variable XNBC_BIN_DIR_TOOLS must be set to this directory.

uid files must be stored /usr/local/lib/uid or in the current directory, or in another directory named in the environment variable XNBC_UID_DIR.

The following environment variables can be set by sourcing the xnbc_env_private or xnbc_env_system_wide, that you should adapt to your environment (for example, xnbc_xterm.dec and xnbc_xterm.sun give a different definition of the XNBC_BIN_REF_XTERM variable).

- XNBC_BIN_DIR_TOOLS directory of xnbc binaries
- XNBC_HELP_DIR directory of xnbc help texts
- XNBC_UID_DIR directory of xnbc uid
- XNBC_BIN_REF_XTERM where is the X terminal emulator you want to use (for analysis tools)
- XNBC_BIN_REF_XTERM where is the X terminal emulator you want to use (for analysis tools)
- XNBC_BIN_REF_PSVIEWER where is the PostScript viewer you want to use
- XNBC_DRUG_DIR directory of xnbc drug files
- XNBC_NEURON_DIR directory of xnbc drug files
- XNBC_BROWSER your web browser (with its path)
- XNBC_ROOT_HTML directory of xnbc html manual

For example, on a Ultrix or Digital Unix machine:

```
setenv XNBC\_BROWSER "/usr/local/netscape/netscape"
setenv XNBC\_BIN\_DIR\_TOOLS "/usr/local/bin/nbc.bin/V8/" setenv
XNBC\_BIN\_REF\_XTERM "/usr/bin/X11/dxterm" setenv
XNBC\_BIN\_REF\_PSVIEWER "/usr/bin/X11/dxvdoc" etc...
```

For both modes, run the xnbc.install script in `"TOP"/XNBC8`. All compilation results are put in `xnbc_install.log`.

Expect to wait about 20 min if you use a 40 MIPS machine (DECstation 5000/240, or RS 6000) 6 min on an Alpha machine (AlphaStation 600/233), 3 min on a Pentium II/333 under Linux.

If you have any problem during installation, please mail us the part of the `xnbc_install.log` file where the problem has been detected.

XNBC was build on the following Unix systems, and should run, provided you have an ANSI C compiler and the Motif development kit:

- Ultrix,
- Digital Unix (OSF1),
- Sun OS,
- Linux,
- AIX,
- HPux.

12.4 User manual

A 140+ page user manual in html format is readable using any web browser. Each chapter can be printed separately (one for each XNBC tool). A color printer will allow to beneficiate from the screens snapshots in color.

The chapter of the manual describing the Conductance based model editor describes G_neuron version 7.14 while the version distributed here is V8.0. It has exactly the same functionality than 7.14 and many more, and few bugs less (this why it is distributed here). The new functionalities (the receptor and transmitters menus allowing to have several types of EPSPs on one neuron, according to the transmitter) is fully implemented -while not extensively tested...- for one neuron in G_neuron V8, but not yet included in the simulator itself. This will be added in XNBC V9. The full manual (in html, readable with any Web browser) is in the directory XNBC8/man/ and in the directory set by XNBC_ROOT_HTML

It is automatically installed. To re-install it, run the manual_install script. You can edit it to choose the directory where to put the file.

A full user paper manual is available on the same ftp site (`ftp://ftp.jussieu.fr/pub/XNBC/xnbc_manual_8_2`

It is too large to be included in the same tar file because of the many screen captures.

The manual is also available on the Web at:

`http://www.b3e.jussieu.fr/logiciels/xnbc8_manual/`

12.5 Using XNBC

This version is 8th.

In this version, the following tools have a full graphic interface:

- the general control program
- the phenomenologic neuron editor
- the Hodgkin-Huxley neuron editor
- the simple network editor
- the full featured network editor
- the drug file editor
- the simulator
- the visualization tool
- the times series analysis tool
- the cluster activity analysis tool

The following tools have an character interface:

- the tool to plot the global network activity
- the tool to plot intracellular recordings
- the tool to plot unit activity in 3D

They are left because they allow to produce more versatile figures than the print menus of the programs with a graphic interface.

If the `/usr/local/bin` directory is in your path, to run xnbc, it should be enough to

```
> setenv DISPLAY my_machine:0.0 xnbc &
```

If you have a specific environment:

```
> setenv DISPLAY my_machine:0.0 source xnbc_env_private xnbc8
```

Then, follow the graphs and on line help...

A kind of tutorial will be found in doc directory as well as several user manuals of the individual tools. Nevertheless, the html manual is now the more frequently updated manual.

12.6 Example data set

For your convenience, an example of the simulation of 2 networks is given in `XNBC8/example_nbc`. To run it, `cd ..` from the building directory (`XNBC8`) (i.e. go to its parent directory) an run xnbc. You will have a predefined small 50 neuron network (20 + 30), that you can simulate, visualize and analyze. Since pre-simulated files are provided, you could visualize and analyze it before any thing else in order to familiarize with these tools.

Examples of other neuron files are in the directory `XNBC8/neurons/`

Don't miss to have also a look to the XNBC Web server, in order to be informed of new versions and related papers.

http://www.b3e.jussieu.fr/xnbc8/

XNBC is provided as an Open Source shareware.

Appendix A

Files of XNBC

The following table gives the name, type and content of the xnbc files.

An important file is the invisible `~/.xnbcrc` file, in the home directory. This file always stores the current working xnbc project and directory, the current network file and the current simulation file. It is automatically updated and allows to quit xnbc and running it back with the same simulation environment. Removing this file is not a good idea, since all programs (except neuron editors, since neuron description can be stored anywhere and their location is stored in the network description file .lnk). If this file does not exists, the current path becomes `./noname_nbc` and the other files `./noname_nbc/noname`.

In the table when a file name is called XX and YY, this signifies that all XX (or YY) files have the same XX (or YY) prefix and a different extension and that XX and YY can be different (but can be also identical).

All text files can be edited using any text editor (but the file structure should be understood . . .). Binary files are not editable.

Program	Input files	Output files	Content	Type	PostScript files
xnbc control panel	~/.xnbcrc	~/.xnbcrc	current files		
Neuron editors					
Phenomenologic	*.P_unit	*.P_unit	unit parameters	text	*.P_unit.ps
	*.L_unit	*.L_unit	unit parameters	text	*.L_unit.ps
	*.B_unit	*.B_unit	unit parameters	text	*.B_unit.ps
Conductance	*.G_unit	*.G_unit	unit parameters	text	*.G_unit.ps
					*.G_unit_XY.ps
					*.G_unit_activ.ps
Network editors					
Simple	~/.xnbcrc	YY.lnk	network description	text	
	XX.lnk	YY.clu	clusters used	text	
		YY.len	axon lengths	bin	
		YY.wgt	synaptic weights	bin	
		YY.anat	unit position	bin	
		~/.xnbcrc		text	
Full featured	~/.xnbcrc	YY.lnk	network description	text	YY.ps
	XX.lnk	YY.clu	clusters used	text	
	~/.net_customrc	YY.len	axon lengths	bin	
		YY.wgt	synaptic weights	bin	
		YY.anat	unit position	bin	
		~/.net_customrc	customizations	text	
		~/.xnbcrc		text	
Drug editor	~/.xnbcrc	Default.drug	drug description	text	
Simulator	~/.xnbcrc	YY.pmb	membrane potential	text	
	XX.lnk	YY.sim	global activity	text	
	XX.clu	YY.tms	time series	text	
	XX.x_def	YY.mod	modifications	text	
	*.P_unit	YY.prm	parameters	text	
	*.L_unit	YY.x_def	parameters	bin	
	*.B_unit	~/.xnbcrc		text	
	*.tms				
Visualization	~/.xnbcrc				YY.ps
	XX.lnk				
	XX.anat				
	YY.sim				
Analysis tools					
Time series	~/.xtmsrc	~/.xtmsrc	user defaults	text	
	~/.xnbcrc	YY.acr	autocorrrelogram	text	YY.ps
	YY.tms	YY.ccr	cross correlogram	text	
		YY.hst	post event histogram	text	
		YY.isi	interspike interval	text	
Time series	~/.xnbcrc	YY.acr	autocorrrelogram	text	YY.ps

Appendix B

PUM model equations

B.1 Equations

The temporal evolution of the potential and the threshold of NBC neurons in a network of N neurons takes place according to the following equations:

$$V_i(t) = U(t - T_p^i) + \left[\sum_{j=1}^{N}(W_{ji}\int_{T_p^i}^{t} x_j(r - d_{ji})\, V_{psp}(t - r)\, dr) + B(t)\right] m(t - T_p^i)$$

$$s_i(t) = \mathbf{F}(t - T_p^i,\, s_i(T_p^i) + S_A)$$

$$x_i(t) = \theta[V_i(t) - s_i(t)]$$

where

$$U(t) = U_{rest} + (U(0) - U_{rest})e^{-t/\tau_V}$$

$$V_{psp}(t) = \begin{cases} V_{pspMax}\,(t/\tau_{attack})\, e^{-\frac{t}{\tau_{attack}}+1} & when\ 0 < t < \tau_{attack} \\ V_{pspMax}\,(\frac{t-\tau_{attack}}{\tau_{decay}} + 1)\, e^{-\frac{t-\tau_{attack}}{\tau_{decay}}} & when\ \tau_{attack} < t \end{cases}$$

$$m(t) = 1 - e^{-t/\tau_{shunt}}$$

$$\mathbf{F}(t, K) = s_{rest} + (K - s_{rest})e^{-t/\tau_s}$$

$\theta[y]$ is the Heaviside step function

$$\theta[y] = \begin{cases} 0 & if\ y < 0 \\ 1 & otherwise \end{cases}$$

$U_{rest},\ U(0),\ \tau_V,\ V_{pspMax},\ \tau_{attack},\ \tau_{decay},\ \tau_{shunt},\ s_{rest},\ S_A,\ \tau_s$ are the parameters which are set at the beginning of each simulation.

The threshold equation shows that the value of the threshold of a neuron at a given time t depends on the firing history of the neuron. In fact by replacing $\mathbf{F}$, the following expression for the threshold of a neuron which has fired p spikes, the kth spike happening at time T_k, is obtained:

$$s(t) = s_{rest}(1 + e^{-\frac{t-T_1}{\tau_s}}) + S_A e^{-\frac{t}{\tau_s}} \sum_{k=1}^{p} e^{\frac{T_k}{\tau_s}}$$

B.2 Default values

These are the default values of a Phenomenologic Model as stored in the .P_unit file:

```
ModelType: P
RestPotential: -70.0000
TimeCsteMbPot: 10.0000
Threshold: -65.0000
EpspSize: 0.5000
IpspSize: 0.0000
NoiseMean: 0.0000
NoiseSd: 0.0000
NoiseFilter: 0
Capacity: 1.0000000000
TimeStep: 1.000
PostSpikePotential: -85.0000
Adaptation: 20.0000
E_K: -90.0000
E_Na: 65.0000
PosReversalPot: 0.0000
NegReversalPot: -90.0000
TimeCsteThreshold: 20.0000
EpspSd: 0.0000
EpspAttack: 2.0000
EpspDecay: 5.0000
IpspSd: 0.0000
IpspAttack: 2.0000
IpspDecay: 5.0000
MembraneShunt: 20.0000
FisterResistance: 10.000
Fatigue: 0.000
LessFatigue: 0
```

IMPORTANT REMARK: If modified with a text editor, remind that the labels are case sensitive.

Appendix C

LIM model equations

C.1 Equations

The temporal evolution of the potential and the threshold of NBC neurons in a network of N neurons takes place according to the following equations:

$$V_i(t) = V_i(t-1)(1 - (\frac{\Delta t}{\tau_V} + \frac{\Delta t \times U_{rest}}{\tau_V})$$

$\theta[y]$ is the Heaviside step function

$$\theta[y] = \begin{cases} 0 & if\ y < 0 \\ 1 & otherwise \end{cases}$$

$U_{rest},\ U(0),\ \tau_V,\ V_{pspMax},\ s_{rest},\ S_A,\ \tau_s$ are the parameters which are set at the beginning of each simulation.

The threshold equation shows that the value of the threshold of a neuron at a given time t depends on the firing history of the neuron. In fact by replacing $\mathbf{F}$, the following expression for the threshold of a neuron which has fired p spikes, the kth spike happening at time T_k, is obtained:

$$s(t) = s_{rest}(1 + e^{-\frac{t-T_1}{\tau_s}}) + S_A e^{-\frac{t}{\tau_s}} \sum_{k=1}^{p} e^{\frac{T_k}{\tau_s}}$$

C.2 Default values

These are the default values of a Leaky Integrator Model as stored in the .L_unit file:

```
ModelType:        L
RestPotential:      - 70.0000
TimeCsteMbPot:        10.0000
Threshold:          - 65.0000
EpspSize:             0.5000
IpspSize:             0.0000
NoiseMean:            0.0000
NoiseSd:              0.0000
NoiseFilter:          0
Capacity:             1.0000000000
```

```
TimeStep:              1.000
PostSpikePotential: - 85.0000
Adaptation:           20.0000
E_K:                 -90.0000
E_Na:                 65.0000
PosReversalPot:        0.0000
NegReversalPot:      -90.0000
TimeCsteThreshold:    20.0000
EpspSd:                0.0000
IpspSd:                0.0000
FilterResistance:     10.000
Fatigue:               0.000
LessFatigue:           0
```

IMPORTANT REMARK: If modified with a text editor, remind that the labels are case sensitive.

Appendix D

BUM model equations

D.1 Equations

These are the same as those of the Phenomenologic model but a second threshold (bursting threshold) inducing a bust. The intra burst frequency is adjusted according to the parameters chosen. When an IPSP arrives the frequency is divided by the factor chosen.

D.2 Default values

These are the default values of a Bursting Unit Model as stored in the .B_unit file:

```
ModelType: B
RestPotential: -70.0000
FreqSpike: 30.0000
FreqMax: 150.0000
ComposeFact: 2.0000
DureeBurstMin: 50.0000
DureeBurstMax: 150.0000
ThresholdBurst: -55.0000
BurstPostSpike: -60.0000
TimeCsteMbPot: 10.0000
Threshold: -65.0000
EpspSize: 0.5000
IpspSize: 0.0000
NoiseMean: 0.0000
NoiseSd: 0.0000
NoiseFilter: 0
Capacity: 1.0000000000
TimeStep: 1.000
PostSpikePotential: -85.0000
Adaptation: 20.0000
E_K: -90.0000
E_Na: 65.0000
PosReversalPot: 0.0000
```

```
NegReversalPot: -90.0000
TimeCsteThreshold: 20.0000
EpspSd: 0.0000
EpspAttack: 2.0000
EpspDecay: 5.0000
IpspSd: 0.0000
IpspAttack: 2.0000
IpspDecay: 5.0000
MembraneShunt: 20.0000
FisterResistance: 10.000
Fatigue: 0.000
LessFatigue: 0
```

IMPORTANT REMARK: If modified with a text editor, remind that the labels are case sensitive.

Appendix E

CBM model equations

E.1 Equations

The following equations are used, where V is the instantaneous membrane potential and E the voltage for which the sigmoid has its inflection point (see below):

The A coefficient is the slope of the sigmoid of the voltage versus the probability of open or closed channels. probability is 0.5 (thus it define the position of the curve on the x axis).

In the following equations, m is associated with activation, while h is associated with inactivation.

p is the exponent for activation and q is the exponent for inactivation

A_m is the slope for activation and A_h is the slope for inactivation.

v_m is the position for activation and v_h is the position for inactivation.

Delayed rectifier K^+ current

$$g = G \times m^p$$

$$\frac{dm}{dt} = \frac{m_\infty - m}{\tau_m}$$

$$\alpha_m = \mathsf{e}^{A_m(V-E)}$$

$$\beta_m = \mathsf{e}^{-A_h(V-E)}$$

$$\tau_m = \frac{1}{\lambda(\alpha_m + \beta_m)}$$

$$m_\infty = \frac{\alpha_m}{\alpha_m + \beta_m}$$

Simple fast Na^+ current

It is implemented using the Michaelis-Menton method.

$$g = G \times {m_\infty}^p \times (1 - m_K)$$

$$m_\infty = \frac{1}{1 + \mathsf{e}^{-2A_m(V-E)}}$$

Hodgkin-Huxley fast Na^+ current

$$g = G \times m^p \times h^q$$

$$\frac{dh}{dt} = \frac{h_\infty - h}{\tau_h}$$

$$m_\infty = \frac{1}{1 + \mathsf{e}^{-2A_m(V-E)}}$$

$$h_\infty = \frac{1}{1 + \mathsf{e}^{-2A_h(V-E)}}$$

$$\tau_h = \text{ constant}$$

AHP K^+ current

$$g = G \times m_\infty$$

$$m_\infty = \frac{[\mathsf{Ca}^{++}]_i}{[\mathsf{Ca}^{++}]_i + K}$$

A K^+ current

$$g = G \times {m_\infty}^p \times h^q$$

$$\frac{dh}{dt} = \frac{h_\infty - h}{\tau_h}$$

$$m_\infty = \frac{1}{1 + \mathsf{e}^{-2A_m(V-E)}}$$

$$h_\infty = \frac{1}{1 + \mathsf{e}^{-2A_h(V-E)}}$$

$$\tau_h = \text{ constant}$$

m K^+ current

$$g = G \times m$$

$$\frac{dm}{dt} = \frac{m_\infty - m}{\tau_m}$$

$$\alpha_m = \mathsf{e}^{A(V-E)}$$

$$\beta_m = \mathsf{e}^{-A(V-E)}$$

$$\tau_m = \frac{1}{\lambda(\alpha_m + \beta_m)}$$

$$m_\infty = \frac{\alpha_m}{\alpha_m + \beta_m}$$

C K^+ current

$$g = G \times m$$
$$\frac{dm}{dt} = \frac{m_\infty - m}{\tau_m}$$
$$\alpha_m = [\mathsf{Ca}^{++}]\mathsf{e}^{A(V-E)}$$
$$\beta_m = \mathsf{e}^{-A(V-E)}$$
$$\tau_m = \frac{1}{\lambda(\alpha_m + \beta_m)}$$
$$m_\infty = \frac{\alpha_m}{\alpha_m + \beta_m}$$

h K^+ current

$$g = G \times m$$
$$\frac{dm}{dt} = \frac{m_\infty - m}{\tau_m}$$
$$\alpha_m = \mathsf{e}^{A(V-E)}$$
$$\beta_m = \mathsf{e}^{-A(V-E)}$$
$$\tau_m = \frac{1}{\lambda(\alpha_m + \beta_m)}$$
$$m_\infty = \frac{\alpha_m}{\alpha_m + \beta_m}$$

Persistent Na^+ current

$$g = G \times m$$
$$\frac{dm}{dt} = \frac{m_\infty - m}{\tau_m}$$
$$\alpha_m = \mathsf{e}^{A(V-E)}$$
$$\beta_m = \mathsf{e}^{-A(V-E)}$$
$$\tau_m = \frac{1}{\lambda(\alpha_m + \beta_m)}$$
$$m_\infty = \frac{\alpha_m}{\alpha_m + \beta_m}$$

High Voltage Activated Ca^{++} current

$$g = G \times m^p \times h$$
$$\frac{dm}{dt} = \frac{m_\infty - m}{\tau_m}$$
$$m_\infty = \frac{1}{1 + \mathsf{e}^{-2A(V-E)}}$$
$$h = \frac{K}{K + [\mathsf{Ca}^{++}]_i}$$

Low Voltage Activated Ca^{++} current

$$g = G \times m^p \times h$$

$$\frac{dm}{dt} = \frac{m_\infty - m}{\tau_m}$$

$$\frac{dh}{dt} = \frac{h_\infty - h}{\tau_h}$$

$$\alpha_m = \mathsf{e}^{A(V-E)}$$

$$\beta_m = \mathsf{e}^{-A(V-E)}$$

$$\tau_m = \text{ constant}$$

$$m_\infty = \frac{\alpha_m}{\alpha_m + \beta_m}$$

$$\alpha_h = \mathsf{e}^{-A(V-E)}$$

$$\beta_h = \mathsf{e}^{A(V-E)}$$

$$\tau_h = \frac{1}{\lambda}$$

$$h_\infty = \frac{\alpha_h}{\alpha_h + \beta_h}$$

NMDA current

$$g = \frac{G}{1 + \eta[\mathsf{Mg}^{++}]\mathsf{e}^{-\gamma V}}$$

E.2 Default values

These are the default values of a Conductance Based Model as stored in the .G_unit file:

```
diametre: 20.00000000000000
TauDecayCa: 20.00000000000000
spike_slope:       10000.000
TimeStep: 0.10000000149012
Temp: 20.00000000000000
Na_ext: 140.00000000000000
Na_int: 14.00000000000000
K_ext: 4.00000000000000
K_int: 140.00000000000000
E_Ca: 124.00000000000000
E_Cl: -89.00000000000000
E_H: -43.00000000000000
PosReversalPot: -10.00000000000000
NegReversalPot: -90.00000000000000
E_fuite: -50.00000000000000
```

```
g_Na: 120.00000000000000
g_K: 36.00000000000000
g_Ca: 1.00000000000000
g_Cl: 0.00000000000000
g_A: 0.00000000000000
g_M: 0.00000000000000
g_C: 0.00000000000000
g_ahp: 0.00000000000000
g_Nap: 0.00000000000000
g_Cab: 0.00000000000000
g_H: 0.00000000000000
g_fuite: 0.30000001192093
g_Syn_pos: 0.05000000074506
g_Syn_neg: 0.05000000074506
capacite: 1.00000000000000
input_fact: 0.00000000000000
tau_attack_epsp: 2.50000000000000
tau_decay_epsp: 10.00000000000000
tau_attack_ipsp: 2.50000000000000
tau_decay_ipsp: 10.00000000000000
Ca_i: 0.00999999977648
K_ext: 4.00000000000000
NoiseMean: 0.00000000000000
NoiseSd: 0.00000000000000
magnesium: 1.00000000000000
g_NMDA: 1.79999995231628
rev_pot_NMDA: 0.00000000000000
eta_NMDA: 0.33000001311302
gamma_NMDA: 0.05999999865890
tau_attack_NMDA: 80.00000000000000
tau_decay_NMDA: 0.66000002622604
Ca_Influx:    0.0001500000
Ca_Removal:    0.0049999999
Ca_Kd:    1.0000000000
K_Ca_Kd:    0.5000000000
lambda_Ik:    0.0810000002
V_Na:  -31.0000000000
A_Na:    0.0650999993
Pow_Na:    3.0000000000
V_K:  -46.0000000000
A_K:    0.0551000014
Pow_K:    4.0000000000
V_Ca:  -40.0000000000
A_Ca:    0.2000000030
Pow_Ca:    2.0000000000
Tau_Ca:   50.0000000000
V_Ia_A:  -20.0000000000
```

```
A_Ia_A:    0.0209999997
Pow_Ia_A:    1.0000000000
V_Ia_B:  -70.0000000000
A_Ia_B:   -0.1000000015
Pow_Ia_B:    1.0000000000
Tau_Ia_B:   10.0000000000
V_Ic:  -90.0000000000
A_Ic:    0.0500000007
V_Im:  -35.0000000000
A_Im:    0.0500000007
V_Inap:  -56.0000000000
A_Inap:    0.0700000003
Vm_Icab:  -45.0000000000
Am_Icab:    0.1000000015
Pow_m_cab:    3.0000000000
Vh_Icab:  -70.0000000000
Ah_Icab:    0.1000000015
V_Ih:  -75.0000000000
A_Ih:   -0.0900000036
lambda_Ic:    5.0000000000
lambda_Im:    0.0030000000
lambda_Inap:    0.1000000015
lambda_Icab:    0.0049999999
lambda_Ih:    0.0005000000
Tau_m_Cab:   10.0000000000
K_rest:    2.5000000000
 m_ahp_init:    0.0000000000
m _K_init:    0.1599999964
l e_potentiel_init:  -60.0000000000
le_potentiel_E_Na_init:   55.0000000000
le_potentiel_E_K_init:  -72.0000000000
m_Ca_init:    0.0010000000
Ca_i_init:    0.0099999998
h_A_init:    0.0500000007
m_Nap_init:    0.0000000000
m_Cab_init:    0.0000000000
m_Ih_init:    0.0000000000
h_Cab_init:    0.0000000000
```

IMPORTANT REMARK: If modified with a text editor, remind that the labels are case sensitive.

Appendix F

References

F.1 Papers about XNBC or using XNBC

Av-Ron E., Vibert J-F. A model for temporal and intensity coding in insect olfaction by a network of inhibitory neurons. BioSystems, 39: 241-250 (1996)

Geoffrois, E., Edeline, J.-M., Vibert, J.-F., 1994, Learning by Delay Modifications. in: Computation in Neurons and Neural Systems. F. H. Eeckman (eds) (Kluwer Academic Publishers, Boston) pp 133–138

Pakdaman, K., Vibert, J.-F., Boussard, E., Azmy, N., 1996, Single neuron with recurrent excitation: Effect of the transmission delay. Neural Networks 9: 797-818

Pakdaman K., Alvarez F., Diez-Martinez O., Vibert J-F. Single neuron with recurrent connection: response to slow periodic modulation. BioSystems, 40: 133-140 (1997)

Pakdaman K, F. Alvarez, Segundo J.P., Diez-Martìnez O., Vibert J-F. Adaptation prevents discharge saturation in models of single neurons with recurrent excitation. Int. J. of Modelling and Simulation, in press (1996)

Pakdaman K., Pham J., Boussard E, Vibert J-F. How transmission delays and noise modify the simple and large neural networks dynamics. In Bower J. (Ed) **Computational Neurosciences: Trends in Research, 1997**. Plenum, New-York. 435-441 (1997)

Pakdaman K., Grotta-Ragazzo C., Malta C.P., Arino O., Vibert J-F. Effect of Delay on the Boundary of the Basin of Attraction in a System of Two Neurons. Neural Networks. In press (1998)

Pham J., Pakdaman K., Vibert J-F. Simulation of spontaneous activity generation in an excitatory network involved in the control of the respiratory rhythm. In Bower J. (Ed) **Computational Neurosciences: Trends in Research, 1997**. Plenum, New-York. 455-461 (1997)

Pham J., Pakdaman K., Champagnat J., Vibert J-F. Spontaneous Activity in Sparsely Connected Excitatory Neural Networks: Effect of Connectivity. Neural Networks. In press (1998)

Segundo, J.P., Stiber, M., Altshuler, E., Vibert, J.-F. Transients in inhibitory driving of neurons and their postsynaptic consequences. Neuroscience 62, 459–480 (1994)

Segundo, J.P., Stiber, M., Vibert, J.-F., Hanneton, S. Periodically modulated inhibition and its post-synaptic consequences. II. Influence of pre-synaptic slope, depth, range, noise and of post-synaptic natural discharges. Neuroscience 68, 693–719 (1995)

Segundo, J.P., Vibert, J.F., Stiber, M., Hanneton, S. Periodically modulated inhibition and its postsynaptic consequences. I. General features. Influence of modulation frequency. Neuroscience 68, 657–692 (1995)

Segundo J.P., Vibert J-F, Stiber M. Periodically modulated inhibition of pacemaker neurons. III. the heterogeneity of the postsynaptic spike train, and how control parameters affect it. Neuroscience. In press (1998).

Vibert, J.-F., Azmy, N. Simulation de la genèse d'un rythme biologique par des réseaux interconnectés. in: Neural networks and their applications. J. Hérault (eds) (EC2, Paris) pp 317–330 (1989)

Vibert, J.-F., Azmy, N. Neuro-bio-clusters: A tool for interacting biological neural networks simulation. in: Artifial Neural Networks. T. Kohonen, K. Makisara, O. Simula, J. Kangas (eds) (Elsevier S.P., NorthHolland Pub, Amsterdam) pp 551–556 (1991)

Vibert, J.-F., Pakdaman, K., Azmy, N. Inter-neural delay modification synchronizes biologically plausible neural networks. Neural Networks 7, 589–607 (1994)

Vibert, J.-F., Pakdaman, K., Cloppet, F., Azmy, N. NBC: a workstation for biological neural network simulation. in: Neural Network Simulation Environments. J. Skrzypek (eds) (Kluwer Academic Publishers, Boston) pp 113–133 (1994)

Vibert, J.-F., Pakdaman, K., Boussard, E., Av-Ron, E. Computational neuroscience and neurology. Nature-Medecine 1, 1247–1248 (1995)

Vibert J-F., Pakdaman K., Boussard E., Av-Ron E. XNBC: A simulation tool. Application to the study of neural coding using hybrid networks. BioSystems, 40: 211-218 (1997)

Vibert, J.-F., Pham, J., Pakdaman, K., Azmy, N., 1995b, XNBC: A simulation tool for neurobiologists. in: The neurobiology of Computation. J. Bower (ed.) (Kluwer Academic Publishers, Boston) pp 346–352.

Vibert J-F., Pakdaman K., Boussard E., Av-Ron E. XNBC: A simulation tool. Application to the study of neural coding using hybrid networks. BioSystems, 40: 211-218 (1997)

Pham J., Pakdaman K., Champagnat J., Vibert J-F. Activity in sparsely connected excitatory neural networks: effect of connectivity. Neural Networks. 11: 415-434 (1998)

Segundo J.P., Vibert J-F, Stiber M. Periodically modulated inhibition of pacemaker neurons. III. the heterogeneity of the postsynaptic spike train, and how control parameters affect it. Neuroscience. 87: 15-47 (1998).

Vibert J-F,, Alvarez F., Pham J. Effects of transmission delays and noise in recurrent excitatory networks. BioSystems, in press (1998).

Pham J., Pakdaman K., Vibert J-F. A discrete map for the dynamics of recurrent excitatory neural networks. BioSystems, in press (1998).

Vibert J-F., Boussard E, Pakdaman K, Av-Ron E. Computational Neuroscience: tools for studying the nervous system. In: Fogelman-Soulié F., Gallinari P. (Eds.) ICANN'95, Proceedings of the International Conference on Artificial Neural Networks, Session 5, Medecine, pp. 63-71 (1995)

Vibert J-F., Pakdaman K, Boussard E. Effects of transmission delays and noise in recurrent excitatory neural networks. Biological Complexity. E. Mizraji, L. Acerenza, F. Alvarez, A. Pomi Eds. DI.R.A.C, Facultad de Ciencias, UdelaR, Montevideo. 97-110 (1997).

Pakdaman K., Pham J., Boussard E, Vibert J-F. How transmission delays and noise modify the simple and large neural networks dynamics. In Bower J. (Ed) Computational Neurosciences: Trends in Research, 1997. Plenum, New-York. 435-441 (1997)

Pham J., Pakdaman K., Vibert J-F. Simulation of spontaneous activity generation in an excitatory network involved in the control of the respiratory rhythm. In Bower J. (Ed) Computational Neurosciences: Trends in Research, 1997. Plenum, New-York. 455-461 (1997)

Vibert J-F., Boussard E, Pakdaman K, Av-Ron E. Computational Neuroscience: tools for studying the nervous system. In: Fogelman-Soulié F., Gallinari P. (Eds.) Industrial Applications of Neural Networks. World Scientific Publishing (Singapor) 439-449 (1998)

Bibliography

F.1 References cited in the text

[1] Av-Ron, E., Parnas, H., Segel, L.A., 1991, A minimal biophysical model for an excitable and oscillatory neuron. **Biol. Cybern.** 65, 487–500.

[2] Hodgkin, A.L., Huxley, A.F., 1952, A quantitative description of membrane current and its application to conduction and excitation in nerve. **J. Physiol. Lond.** 117, 500–544.

[3] MacGregor, R.J., 1987, **Neural and Brain Modeling**. Academic Press, San Diego.

[4] Neurotoxins (second Edition) June 1996, Suppelement of **Trends in Neurosciences**. Elsevier Trends Journals.

[5] Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery B.P., 1992, **Numerical Recipes in C**. The art of Scientific Computing. Second Edition. Cambridge Univ. Press.

[6] Vibert, J.-F., Pakdaman, K., Cloppet, F., Azmy, N., 1994b, NBC: a workstation for biological neural network simulation. in: **Neural Network Simulation Environments**. J. Skrzypek (eds) (Kluwer Academic Publishers, Boston) pp 113–133.

[7] Vibert J-F., Pakdaman K., Boussard E., Av-Ron E. XNBC: A simulation tool. Application to the study of neural coding using hybrid networks. **BioSystems**, 40: 211-218 (1997)