1

NEXUS User Manual

beta release - version 0.8

1992

Paul Sajda, Ko Sakai, and Leif H. Finkel

N00014-90-J-1864

Department of Bioengineering and
Institute of Neurological Sciences
University of Pennsylvania
Philadelphia, PA. 19104-6392

Contents

Statement A per telecon
Dr. Harold Hawkins ONR/Code 1142
Arlington, VA 22217-5000

NWW 3/31/92

1 The NX File

1.1 NX File Specifications

NX is the Network Architectural Specification Language used for defining network structure and connectivity in NEXUS. Creating an NX file consists of defining a set of parameters which 1) describe the functional properties and spatial layout of networks of cells (or units) and 2) specify the spatial and weighting properties of the cells' connection fields. The limited syntax of the NX language makes it easy to learn–creating new architectures requires no more than editing an existing NX file and changing the relevant parameters.

The naming convention for an NX file is either *filename* or *filename*.nx. Naming conventions are used throughout NEXUS and it is highly recommended that they be followed.

1.2 NX Syntax–an example

```
/* an example of an NX file */
Network Contour {
                # of units       = 4096;
                x dimension      = 64;
                y dimension      = 64;
                transfer function = sigmoid (0,100,1.0);
                threshold        = const(1.0);
                decay            = 1.0;
                clamp            = off;
                initial firing rate = const(0.7);
                evaluation per cycle = 1;
                scale            = 1.0;
                offset           = 0.0;
                x position       = 100.0;
                y position       = 100.0;

                connections {  /* this is a comment */
                       from Input {
                                projection       = aoi(5,5,59,59);
                                mapping type     = direct;
                                connection field shape = ellipse;
                                length           = 5;
                                width            = 5;
                                rotation angle   = 0;
                                shift x          = 0;
                                shift y          = 0;
                                weight function  = file(file_name);
                                feedback         = off;
                                }
               }
}
```

Segment of NX code.

1.2.1 Network Definitions

Variables names are in **bold** and variable values are in *italics*.

1. **Network** *network_name*
   Each network should be given a unique name, consistent throughout the NX file, since the user will reference this name when establishing and modifying network connections and changing parameters within NEXUS. The network name is a single alphanumeric string (a to z, A to Z, 0 to 9) and may contain underscores (_) but no spaces. For the remainder of this section we will use the name "current_network" to refer to this network.

2. **# of units** = *number_of_cells*
   Defines the total number of cells in the network. One may use "number" in place of "#". The variable takes on integer values and should be defined immediately after the network name.

3. **x dimension** = *number_of_columns_in_network*
   y dimension = *number_of_rows_in_network*
   These variables take on integer values and define the spatial layout of the cells in the network. Both these variables should be defined prior to the following sets of parameters.

4. **transfer function** = *function_type_and_arguments*
   This statement defines the transfer function ($T(input)$) of cells in the network. Currently four types of functions are available:

   (a) **sigmoid***(min, max, slope)*
   Defines a sigmoidal transfer function of the form:

   $$output = (max - min) * \frac{1}{1 + e^{-(input - threshold)*slope}} + min$$

   where the threshold is defined below.

   (b) **linear***(min, max, slope)*
   Defines a linear transfer function of the form:

   $$min \leq slope * (input - threshold)) \leq max \quad \Rightarrow output = slope * (input - threshold)$$

   $$min > slope * (input - threshold)) \quad \Rightarrow output = min$$

   $$max < slope * (input - threshold)) \quad \Rightarrow output = max$$

   where the threshold is defined below.

(c) **step**_(min, max)_
Defines a step function of the form:

$$(input \geq threshold)) \Rightarrow output = max$$

$$(input < threshold) \Rightarrow output = min$$

where the threshold is defined below.

(d) **pgn**_(pgn_function_name)_
Defines the user defined PGN function _pgn_function_name_.

The arguments to **sigmoid**, **linear** and **step** functions are optional–if the arguments and parentheses are not included the _min_, _max_ and _slope_ parameters are set to the values specified in the "settings file". When no "settings file" is specified at NEXUS initialization the default values are used (_min_=0.0, _max_=100.0 and _slope_=1.0).

For historical reasons, **simple** and **binary** are also valid transfer functions and are identical to **sigmoid** and **step** respectively, except they do not take arguments (values are set to the defaults in the settings file).

5. **threshold** = _threshold_type_and_arguments_
Threshold function for cells in the network. Currently two functions are supported:

   (a) **const**_(value)_
   Assigns all cells in the network a threshold equal to _value_.

   (b) **rand**_(min,max)_
   Assigns each cell a threshold using a uniformly distributed random variable with values between _min_ and _max_.

   value, _min_ and _max_ should all be specified as floating point values.

6. **decay** = _decay_coefficient_
Value which the total input (sum of voltages) is multiplied by before passing through the transfer function ($T(input)$):

$$output = T(input) * decay\_coefficient$$

_decay_coefficient_ should be specified as a floating point value.

7. **clamp** = _on_or_off_
A network can be clamped (set so that it is not explicitly evaluated) by turning its clamp state "on". Input networks are usually clamped "on".

8. **initial firing rate** = _function_
Assigns the initial firing rates of the cells in the network. Currently three functions are supported:

   (a) **const**_(value)_
   Assigns all cells in the network a firing rate equal to _value_.

(b) **rand**(*min,max*)
Assigns each cell a firing rate using a uniformly distributed random variable with values between *min* and *max*.

(c) **file**(*filename*)
Assigns the cells' firing rates using the values specified in the file *filename*.

value, *min* and *max* should all be specified as floating point values. *filename* is a string representing the name of the file to load. The default directory for the file is the current directory, however, one may add the full directory name if the specified file is not in the current directory

9. **evaluation per cycle =** *value*
   Sets the number of times the network should be evaluated for each simulation cycle. *value* should be specified as an integer.

10. **scale =** *value*
    offset = *value*
    These parameters modify the output (firing rate) of the network in the following manner:

$$output_{modified} = (scale * output) + offset$$

default values are scale = 1.0 and offset = 0.0. *values* should be specified as floating point numbers.

11. **x position =** *value*
    y position = *value*
    Defines the location of the network on the display in world coordinates. These statements are optional. *value* should be a floating point value.

Order of Variable Definitions
The only restriction on the order of variable definitions is that **# of units**, **x dimension** and **y dimension** should be defined prior to the other parameters.

ERROR Messages
WARNING or ERROR messages will be displayed if an incorrect number of variables or arguments are used. If the order of variable definitions restriction is violated then an ERROR will be displayed. WARNINGs result in continuation of the NX build, while ERRORs will halt the build and wait for further user instructions.

Optional Space, Tab, Return, and Semi-colon
A user may place any number of spaces, tabs or returns between variable definitions or between words. However, at least one space, tab, return or semicolon is required to separate variable definitions.

Comments
Comments can be placed anywhere between variable definitions, delimiting the comment with /* and */, analogous to C syntax. Note: Do not use the character *(asterisk) within the comment or the NX file will be incorrectly interpreted.

1.2.2 Connection Field Definitions

Following the Network Definitions, connections fields, if any, are defined. An arbitrary number of connection fields can be defined for a given network. Both retrograde (connections from a different network) and anterograde (connections to a different network) can be specified for each network.

1. **from** or **to** or **pgn-to** *network_name*
   Specifies that a connection is to be made between the current network and *network_name*. **from** indicates a retrograde connection, **to** a anterograde connection, and **pgn-to** specifies that the connection should be treated specially so that it can be accessed by pgn functions. The network *network_name* is classified as the target network, regardless of whether the connection is retrograde or anterograde.

2. **projection** = *projection_type*
   This parameter defines the region of cells in the current network having this connection field definition. projection = "full" indicates that all cells in the current network should make the connections specified by the connection field. If projection = "aoi(xll,yll,xur,yur)" then only those cells in the Area of Interest, defined by the rectangle having lower-left coordinates (xll,yll) and upper-right coordinates (xur,yur), should make the connections defined in the connection field. If overlapping AOIs are defined, then a larger (in area) AOI must be defined before a smaller one. Note: Currently, AOI defined connections cannot be interactively changed within NEXUS–only projections of type "full" can be changed.

3. **mapping type** = *map_type*
   Specifies how connections should be mapped between the networks. If *map_type* is set to "direct" then the locations of the target cells is made relative to the dimensions of the current network. For example, if the current network is half the size of the target network then the target locations would begin at the lower left of the target network, and would span an area equal to the size of the current network. If *map_type* is set to "normalize" then the location of the target projection is normalized against the ratio of the target to current network size, forcing the target locations to span the area of the target network. If both the current and target network have the same dimensions then "direct" and "normalize" produce identical results.

4. **connection field shape** = *rf_type*
   Defines the mask shape for the connection field. *rf_type* may take on the values "ellipse" or "rectangle".

5. **length** = *value*
   width = *value*
   Defines the size of the connection field having the shape *rf_type*. *value* should be specified as an integer.

6. **rotation angle** = *value*
   shift x = *value*
   shift y = *value*
   Defines the specific spatial transformations for the connection field projection. **rotation angle** specifies that the connection field should be rotated *value* degrees, where 0^o is vertical, 90^o is horizontal, with angles increasing clockwise. **shift x** and **shift y** specify a relative shift of all connection field target projections. For example, for the case of two networks of the same size, if the current network has a connection field with **shift x** = 1

and **shift y** $= 2$ then cell (i, j) in the current network projects to the cell $(i+1, j+2)$ in the target network.

7. **weight function** = *weight_function_type_and_arguments*
   Defines the function used to specify the weights of the connections in the connection field. Currently the following weight functions are supported:

   (a) **const***(value)*
   Assigns all connections a weight equal to *value*.

   (b) **rand***(min,max)*
   Assigns each connection a weight using a uniformly distributed random variable with values between *min* and *max*.

   (c) **file***(filename)*
   Assigns weights using the values specified in the file *filename*.

   (d) **exp***(max, min, σ)*
   Assigns weights using a two dimensional exponential function. *max* and *min* are the maximum and minimum values of the weights and σ is the space constant of the exponential:

   $$(max - min) * e^{\frac{\sqrt{(x^2+y^2)}}{\sigma}} + min$$

   and where x and y are the spatial position of the target cells.

   (e) **dog***(σ_{ex},S_{ex},σ_{in},S_{in})* Assigns weights using a difference–of–gaussians function. $(\sigma_{ex},\sigma_{in})$ and (S_{ex},S_{in}) are the standard deviations and scaling factors for the excitatory and inhibitory gaussian lobes:

   $$\frac{1}{2\pi}\left[S_{ex}\left(\frac{1}{\sigma_{ex}}e^{\frac{-x^2}{2\sigma_{ex}}}\right)\left(\frac{1}{\sigma_{ex}}e^{\frac{-y^2}{2\sigma_{ex}}}\right) - S_{in}\left(\frac{1}{\sigma_{in}}e^{\frac{-x^2}{2\sigma_{in}}}\right)\left(\frac{1}{\sigma_{in}}e^{\frac{-y^2}{2\sigma_{in}}}\right)\right]$$

   and where x and y are the spatial position of the target cells.

   (f) **line***(σ_{ex},S_{ex},σ_{in},S_{in})* Assigns weights using a one dimensional (oriented) difference–of–gaussians (useful for detecting line orientation). $(\sigma_{ex},\sigma_{in})$ and (S_{ex},S_{in}) are the standard deviations and scaling factors for the excitatory and inhibitory gaussian lobes:

   $$\frac{1}{\sqrt{2\pi}}\left[S_{ex}\left(\frac{1}{\sigma_{ex}}e^{\frac{-x^2}{2\sigma_{ex}}}\right) - S_{in}\left(\frac{1}{\sigma_{in}}e^{\frac{-x^2}{2\sigma_{in}}}\right)\right]$$

   and where x is the horizontal spatial coordinate of the target cells.

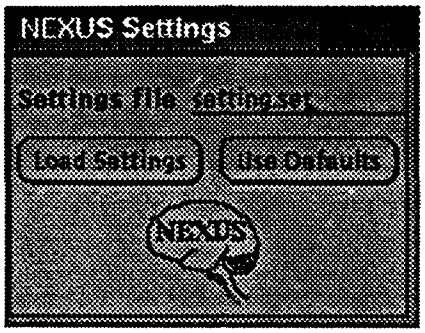
8. **feedback** = *on_or_off*
   Defines whether the cell should connect to itself. Valid only if the current network equals the target network and the connection field overlaps the cell's position.

Order of Variable Definitions
The only restriction is that **projection** should be defined prior to the other parameters. When a user defines overlapping AOIs, the larger AOI must be defined before than smaller one (i.e., if one wants to define a "full" connection and an "aoi" connection, the "full" connection must be defined first.

2 NEXUS Menus

A particularly attractive feature of NEXUS is that it consists of an easy–to–use interactive graphics display for testing and simulating models constructed with NX. The following section will describe the menu functions available in NEXUS

2.1 NEXUS Settings



Settings file: settings.set

The settings menu allows the user to specify certain default parameters and display variables at startup. The values of the parameters are located in a "settings file". The following is an example of a settings file:

```
/* settings file for NEXUS */
file: nx_test
file_notes: nx_test.notes
max_firing: 100.0
min_firing: 0.0
slope: 1.0
conductance_max: 3.0
conductance_min: -3.0
pos_x: 0.0
pos_y: 0.0
pos_z: 0.0
text_width: 30.0
text_height: 50.0
```

Currently the order of parameter definition must be followed exactly and all parameters must be specified in the file. The naming convention for settings files is *filename*.set. The following are the parameters which are specified in the settings file:

1. The first line can be a comment, using the same syntax and restrictions as comments in NX except comments are additionally restricted to be the first line of the settings file.

2. **file:***nx_file_name*
   This specifies the default NX file to include in the **Build Simulation** and **Load Simulation** menus.

3. **file_notes:***notes_file_name*
   As the simulator is run, user actions are written to the file *notes_file_name*. This is useful if one wants to examine the "history" or event sequence after a simulation.

4. **max_firing:***max_firing_rate*
   min_firing:*min_firing_rate*
   slope:*slope_value*
   Defines the default maximum and minimum firing rates and slope for cells which do not specify these values in their transfer function. In addition, the values *max_firing_rate* and *min_firing_rate* are used by the system to set the scale of the "activity" color legend.

5. **conductance_max:***max_weight*
   conductance_min:*min_weight*
   These parameter are used solely for setting the scale of the "connection" color legend.

6. **pos_x:***x*
   pos_y:*y*
   pos_z:*z*
   Defines default position of viewer, relative to the networks, in world coordinates.

7. **text_width:***dim_x*
   text_height:*dim_y*
   Defines default dimensions of displayed text for network names.

All numeric values in the settings file should be specified as floating point numbers.

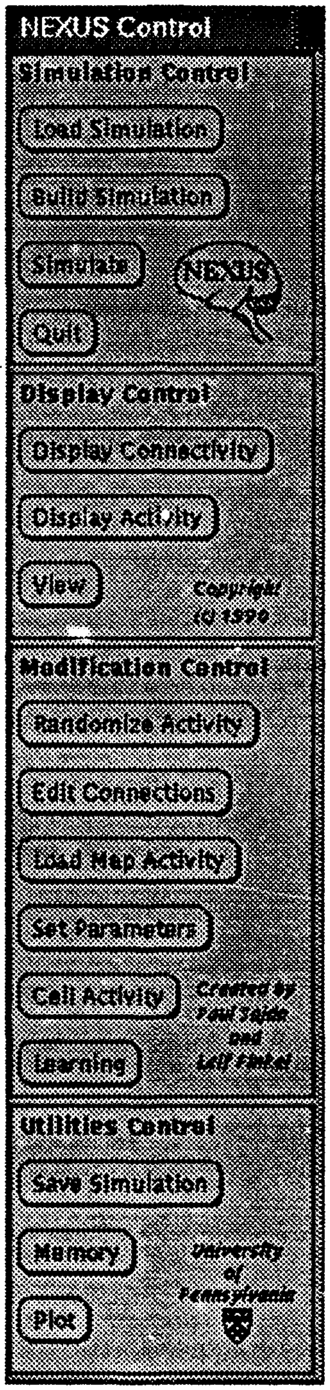
2.1.1 Buttons

1. **Load Settings**
   Loads the parameter values specified in the **Settings file**. (Activates **NEXUS Control** menu.)

2. **Use Defaults**
   Ignores the settings file and loads the system's internal default parameters. The following is a list of the default values: (Activates **NEXUS Control** menu.)

```
file: temp
file_notes: temp.notes
max_firing: 100.0
min_firing: 0.0
slope: 1.0
conductance_max: 1.0
conductance_min: -1.0
pos_x: 0.0
pos_y: 0.0
pos_z: 0.0
text_width: 100.0
text_height: 50.0
```

2.2 NEXUS Control



Once the settings have been loaded, NEXUS begins by popping-up the main display and control windows. Most of the buttons in the control window are associated with additional menus, which are displayed when the button is activated.

2.2.1 Buttons

SIMULATION CONTROL

1. **Load Simulation**
   For loading an existing saved (*filename*.save) simulation. (Activates **Simulation filename** menu.)
2. **Build Simulation**
   For building a simulation using the architecture defined in an NX file. (Activates **Simulation filename** menu.)
3. **Simulate**
   For running the simulation and evaluating the currently loaded networks. (Activates **Simulate** menu.)
4. **Quit**
   Exit NEXUS.

DISPLAY CONTROL

1. **Display Connectivity**
   For graphically displaying network connectivity and saving connections to files. (Activates **Connection display** menu.)
2. **Display Activity**
   Display the current activity (firing rate) of the cells in all networks.
3. **View**
   For changing the position of networks on the graphics display. (Activates **Simulation View** menu.)

MODIFICATION CONTROL

1. **Randomize Activity**
   Used for loading random activity patterns into all networks which are not clamped (clamp=off). (Activates **Randomize Cell Activity** menu.)
2. **Edit Connections**
   Edit the values of the weights for cells in a particular network. (Activates **Edit Connections** menu.)
3. **Load Map Activity**
   For loading activity from a user defined file into a particular network. (Activates **Load Activity** menu.)
4. **Set Parameters**
   For changing and examining specific network and cell parameters. (Activates **Set Parameters** menu).

5. **Cell Activity**
   For changing and examining cell activity (firing rate). (Activates **Modify Cell Activity** menu).

6. **Learning**
   Display learning rule options. (Activates **Learning** menu).

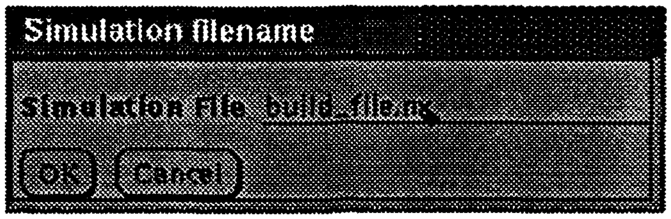
UTILITIES CONTROL

1. **Save Simulation**
   Save the currently loaded simulation. (Activates **Save Simulation** menu).

2. **Memory**
   Display memory and system statistics in an xterm window.

3. **Plot**
   For interactive 3D plotting of network activity. (Activates **Plot Data** menu.)

2.3 Simulation Filename



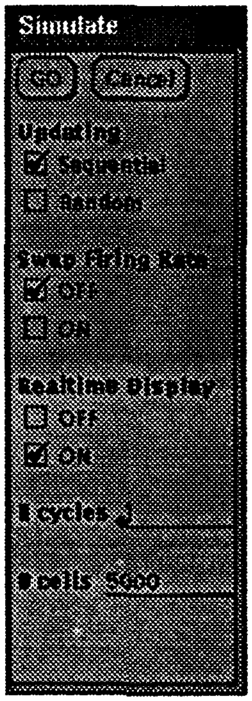
This menu is used to enter the filename for loading or building.

2.3.1 Buttons

1. **OK**
   Load or Build the network specified in the **Simulation File**. If the menu is activated by pressing **Load Simulation** then a *filename*.save file is assumed. If activated by **Build Simulation** then a file containing NX code is expected.

2. **Cancel**
   Cancel the command and quit the menu.

2.4 Simulate



Updating
[✓ Sequential]
Random

Swap Firing Rate
[✓ OFF]
ON

Realtime Display
OFF
[✓ ON]

cycles 1

cells 5000

This menu is used to run the simulation.

2.4.1 Buttons

1. **GO**
   Start simulating the model. The simulation runs for **# cycles**, where the definition of a cycle depends on the type of updating which is chosen.

2. **Cancel**
   Cancel the command and quit the menu.

2.4.2 Check Boxes

1. **Updating**
   Set order of evaluation.

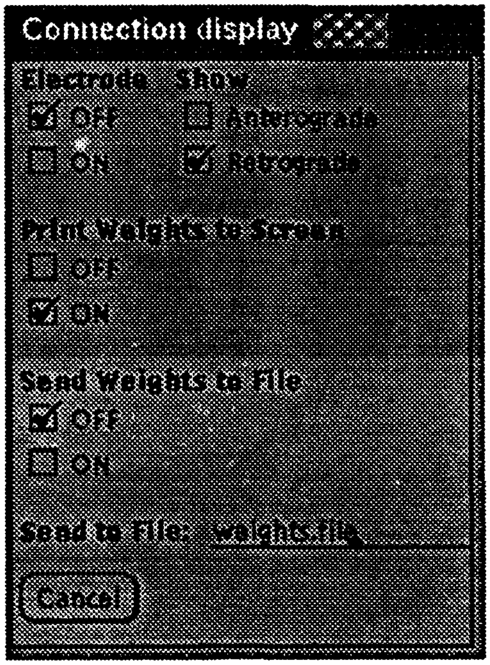
   (a) Sequential – Evaluate cell activity by starting at the first network specified in the NX file and continue sequentially. For sequential updating, one cycle is defined as a single evaluation of all cells in all networks.

   (b) Random – Randomly choose a network and cell for updating. For random updating, one cycle is defined as the evaluation of **# cells**.

2. **Swap Firing Rates**
   When "on" all cells use the old (previous) firing rates of their inputs to determine their current firing rate (should only be set to "on" when using hebbian learning).

3. **Realtime Display**
   When "on" a cell's new firing rate is displayed after it has evaluated. If "off", the system waits until all cycles of the simulation have elapsed and then displays the current firing rates of all cells in all networks. For simulations of multiple cycles, the "off" option increases performance (faster simulations).

2.5 Connection display



Connection display

Electrode Show

[✓ OFF] Anterograde

ON [✓ Retrograde]

Print Weights to Screen

OFF

[✓ ON]

Send Weights to File

[✓ OFF]

ON

Send to File: weights.file

Cancel

This menu allows the user to interactively examine network connections.

2.5.1 Buttons

1. **Cancel**
   Cancel the command and quit the menu.

2.5.2 Check Boxes

1. **Electrode**
   Turning electrode "on" allows the user to interactively examine connections for a particular cell by using the mouse to point to the cell and then clicking the <u>left mouse button</u>. "off" disables the electrode for examining connections.

2. **Show**
   Setting this check box to "Retrograde" causes input connections (all those connections coming into a cell) to be displayed. Currently the "Anterograde" option is not available.[1]

3. **Print Weights to Screen**
   "On" prints a list of the weights in an xterm window.

4. **Send Weights to File**
   "On" sends a list of the cell's weights to the file specified in **Send to File:**. The weights are written when the electrode is positioned over a given cell and the <u>left button</u> is clicked.

[1]Note, a connection defined as anterograde in the NX file will be displayed as a connection for the cells it is connected to, not the cells it is projected from.

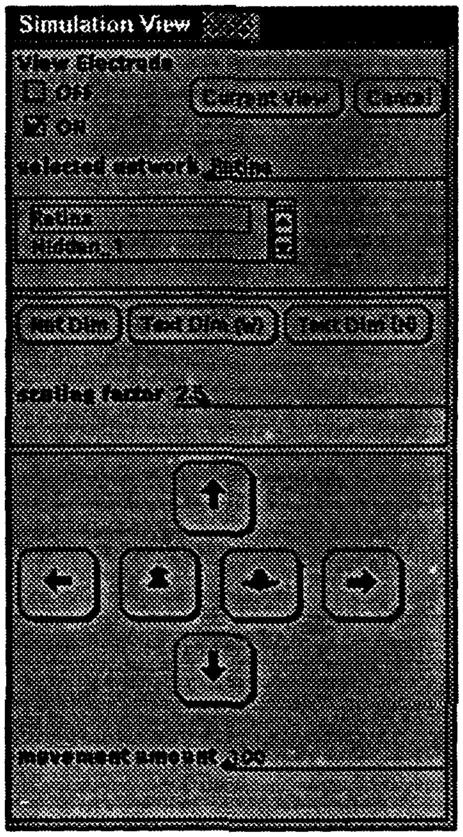
Subsequent presses of the left mouse button with write over the existing weights in the file, therefore one should turn **Send Weight to File** "off" when the desired cell's weights are saved. The format of the saved file is the following:

```
Network <selected_network> from Network <input_network_1>

13 12 -0.393815
12 12 0.103814
11 12 -0.109145
10 12 -0.020929
9 12 0.299897
13 11 -0.198642
12 11 -0.144811
      .
      .
      .
Network <selected_network> from Network <input_network_2>

11 11 -0.253689
10 11 0.340454
9 11 0.318377
13 10 -0.270580
12 10 0.189015
11 10 0.007199
      .
      :
      .
```

where the first two columns represent the x and y coordinates of the input cell and the third column is the value of the weight.[2]

[2] To display these weights using external graphing software (such as *xprism3*) it is advisable that the saved file be run through the utility *connect_wt()* so that the weights are properly sorted for surface/mesh display.

2.6 View



View Electrode
Off [✓ On]
selected network Retina
scaling factor 2.5
movement amount 100

This menu allows the user to interactively change the view of the network–includes scaling and translation of networks and associated text.

2.6.1 Buttons

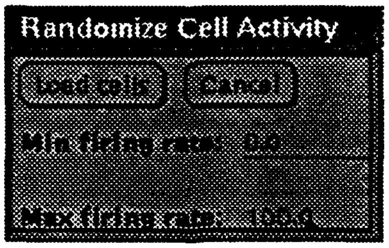
1. **Current View**
   Display the current view and position of the networks.
2. **Net Dim**
   Change the scale (size) of the **selected network** by a factor of **scaling factor**.
3. **Text Dim (W)**
   Change the width of the **selected network**'s text display by a factor of **scaling factor**. Scaling factor can be an integer or floating point number.
4. **Text Dim (H)**
   Change the height of the **selected network**'s text display by a factor of **scaling factor**. Scaling factor can be an integer or floating point number.
5. **(arrows)**
   Pan left, right, up and down by the amount specified in **movement amount**.

6. **Cancel**
   Cancel the command and quit the menu.

2.6.2 Check Boxes

1. **View Electrode**
   Turning electrode "on" allows the user to interactively move and position a network on the display. A network can be selected by double–clicking the left mouse button within the network's associated bounding box. The bounding box will become hilited (filled white). The network can then be positioned by moving the mouse to the new location and clicking the right mouse button. Note that when the right mouse button is clicked, the upper–left corner of the network is positioned at the current mouse position.

2.7 Randomize Cell Activity



Min firing rate: 0.0

Max firing rate: 100.0

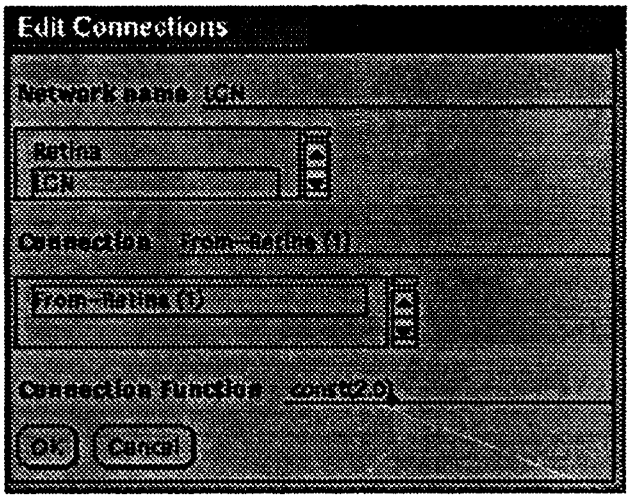
This menu allows the user to load all cells, not clamped “on”, with a random firing rate.

2.7.1 Buttons

1. **Load cells**
   Loads each cell in all unclamped networks with a firing rate determined using a uniformly distributed random variable between **Min firing rate:** and **Max firing rate:**.

2. **Cancel**
   Cancel the command and quit the menu.

2.8 Edit Connections



Network name LGN

Connection From-Retina (1)

Connection Function const(2.0)

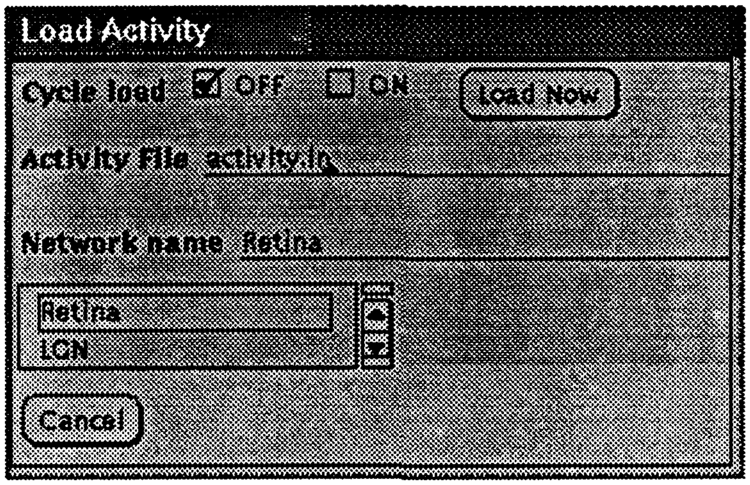
This menu allows the user to interactively change the weight function of the connections in the network.

2.8.1 Buttons

1. **OK**
   Changes the weights of all the cells in network **Network name** for the connection **Connection** using the the function **Connection Function**. The syntax for **Connection function** is identical to that for NX (e.g. "const(2.0)").

2. **Cancel**
   Cancel the command and quit the menu.

2.9 Load Activity



This menu allows the user to set the firing rate of cells in a given network using a matrix of activity values.

2.9.1 Buttons

1. **Load Now**
   Loads the cells in the network **Network name** *using the first file specified in* **Activity File**. **Activity File** is a list of files, with each file containing a matrix of activity values. For example, if a user has defined two matrices of activity values, stored in the files "input.1" and "input.2", then the contents of the file specified as **Activity File** would simple be:

   ```
   input.1 <network_name>
   input.2 <network_name>
   ```

   Clicking **Load Now** would load the values specified in "input.1". `<network_name>` is an optional argument, which if specified overrides the value of **Network name**. The format of the files containing the actual activity matrices is of the form:

   ```
   1.0 1.0 1.0 . . .
   2.0 2.0 2.0 . . .
    .   .   .
    .   .   .
    .   .   .
   ```

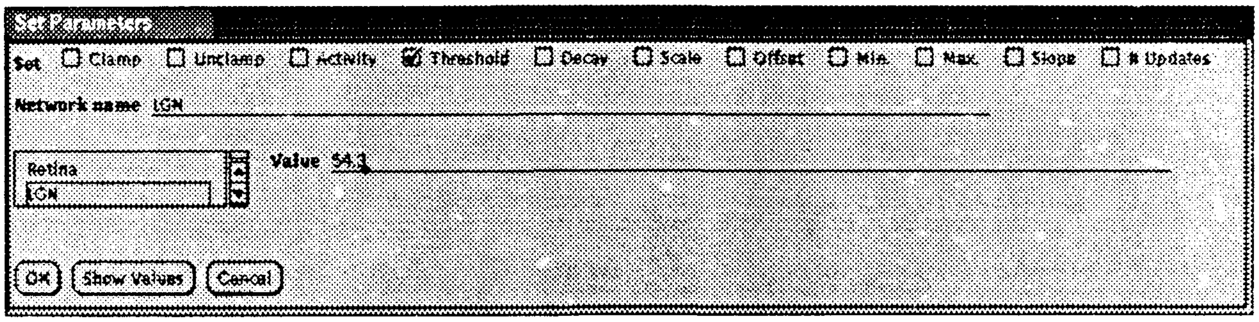
   where the values are in row/column form and the dimensions of the matrix equal the dimensions of the network.

2. **Cancel**
   Cancel the command and quit the menu.

2.9.2 Check Boxes

1. **Cycle load**
   When "on" the system loads the next file in the **Activity file** into the network **Network name** after the end of the currenι ꞇycle. If the simulation lasts for more cycles than there are files in **Activity File** the loading wraps-around and starts with the first file in **Activity File**. If the optional argument **`<network_name>`** is included then this network is loaded with the activity values instead of the network **Network name**.

2.10 Set Parameters


Set Parameters

Set ☐ Clamp ☐ Unclamp ☐ Activity ☑ Threshold ☐ Decay ☐ Scale ☐ Offset ☐ Min. ☐ Max. ☐ Slope ☐ # Updates

Network name LGN

Retina
LGN

Value 54.1

This menu allows the user to set and examine several network parameters.

2.10.1 Buttons

1. **OK**
   Change the parameter specified by **Set** for the network **Network name** to the value **Value**.

2. **Show Values**
   Print the values of the parameters for the network **Network name**.

3. **Cancel**
   Cancel the command and quit the menu.

2.10.2 Check Boxes

1. **Clamp**
   Clamp all cells in the network **Network name** (clamp=on).

2. **Unclamp**
   Unclamp all cells in the network **Network name** (clamp=off).

3. **Activity**
   Set the activity of all cells in the network **Network name** to **Value**

4. **Threshold**
   Set the threshold of all cells in the network **Network name** to **Value**.

5. **Decay**
   Set the decay constant of all cells in the network **Network name** to **Value**.

6. **Scale**
   Set the scale coefficient of all cells in the network **Network name** to **Value**

7. **Offset**
   Set the offset of all cells in the network **Network name** to **Value**.

8. **Min.**
   Set the minimum firing rate of all cells in the network **Network name** to **Value**

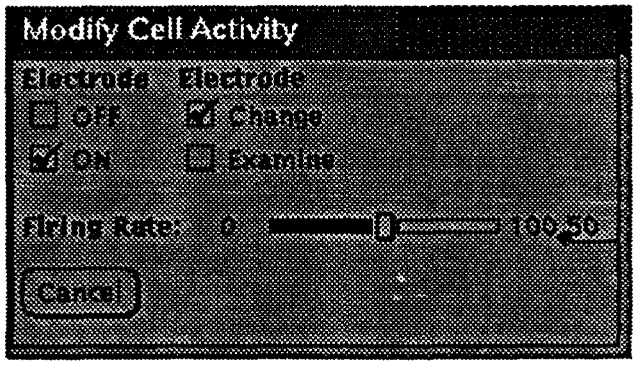
9. **Max.**
   Set the maximum firing rate of all cells in the network **Network name** to **Value**

10. **Slope**
    Set the slope of the transfer function for all cells in the network **Network name** to **Value**.

11. **Slope**
    Set the number of evaluations per cycle for the network **Network name** to **Value**.

(see the section on *NX Syntax–an example* for an explanation of these parameters.

2.11 Modify Cell Activity



This menu allows the user to set and examine individual cell activity.

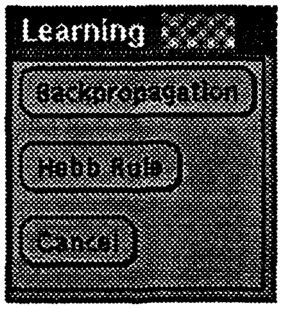
2.11.1 Buttons

1. **Cancel**
   Cancel the command and quit the menu.

2.11.2 Check Boxes

1. **Electrodes**
   When the first **Electrode** is set "on" the mouse is available for "stimulating" and examining cells. When the second electrode is set to "Change", clicking the left mouse button over the position of a particular cell in the display will set the new activity of the cell to the value **Firing Rate**. The value of **Firing Rate** can be entered either through the keyboard or set using the slider bar. When the second electrode is set to "Examine", the parameters of the selected cell are displayed in an xterm window, but cell activity is not changed.

2.12 Learning

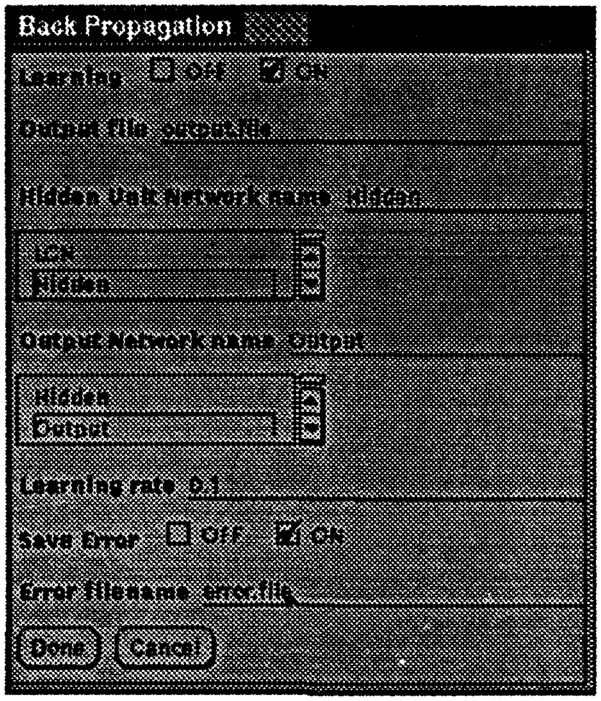


This menu allows the user to access menus for different learning rules.

2.12.1 Buttons

1. **Backpropagation**
   Activate backpropagation menu.
2. **Hebb Rule**
   Activate hebbian learning menu.
3. **Cancel**
   Cancel the command and quit the menu.

2.13 Back Propagation


Back Propagation

Learning [ ] Off [✓ On]

Output file outputfile

Hidden Unit Network name Hidden

LGN
Hidden

Output Network name Output

Hidden
Output

Learning rate 0.1

Save Error [ ] Off [✓ On]

Error filename error.file

This menu allows the user to simulate PDP models using the backpropagation learning algorithm (delta rule).

2.13.1 Buttons

1. **Done**
   Creates intermediate parameters required for backpropagation training and exits menu.

2. **Cancel**
   Cancel the command and quit the menu.

2.13.2 Check Boxes

1. **Learning**
   When learning is "on", backpropagation training is activated. For backpropagation training, the user must specify several parameters: 1) An **Output file** containing a list of filenames. The individual files contain matrices of desired outputs. 2) The **Hidden Unit Network name**, indicating which network is considered the hidden layer. 3) The **Output Network name**, or the network whose output should be compared with the desired outputs in the files from **Output file**. 4) A **Learning rate** coefficient, indicating the magnitude of the weight change. In addition, 5) the user must select a particular network as the input network and use the **Load Activity** menu to cyclically load input activity patterns. For example, consider a three network simulation consisting of the networks "Input", "Hidden" and "Output". We begin by turning backpropagation learning "on"and

setting the **Output file** to the filename "output.out". The contents of "output.out" might be the following:

```
desired_output.1
desired_output.2
desired_output.3
```

where the three files "desired.?" each contain a matrix consisting of the desired outputs for the first three input examples. We then use the Load Activity menu to set cyclic loading of the input (example) files. An example of an input **Activity File** might be "input.in", which would consist of a list of the following files:

```
input.1
input.2
input.3
```

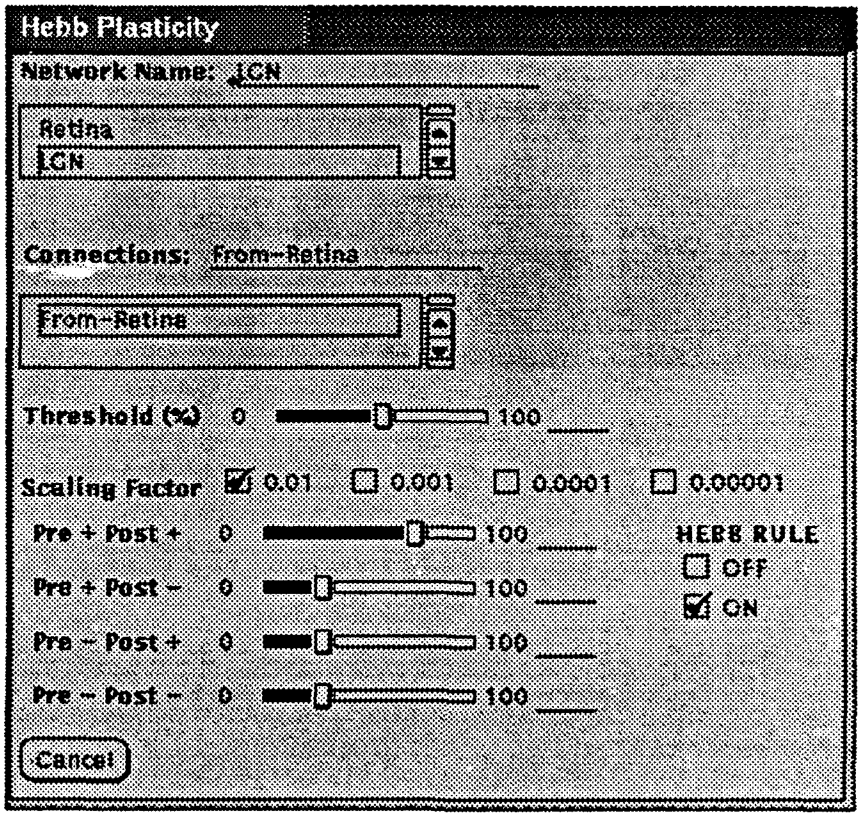
These three files "input.?" each consist of a matrix of values which represent the input activity to the network "Input" (The dimensions of these matrices should match the size of the input network). Note that both the "input.in" and "output.out" should consist of the same number of files–the first file in "output.in" (desired_output.1) corresponds to the desired outputs when the input is the first file in "input.in" (input.1), and so on. Finally, return to the **Back Propagation** menu and set the **Learning rate** and click **Done**. Now, the network can be simulated (use sequential and swap firing rate=off mode), and the weights will be changed according to the learning rule.

2. **Save Error**
   When "on", saves the sum-squared error for each input example to the file **Error filename**. For a simulation of ten cycles and using the example above, 30 floating point values would be sent to the file **Error filename**:

```
<error_input.1_cycle1>
<error_input.2_cycle1>
<error_input.3_cycle1>
<error_input.1_cycle2>
<error_input.2_cycle2>
          .
          .
          .
```

The values in this file can be used to plot the learning curves for the network.

2.14 Hebb Plasticity



Hebb Plasticity

Network Name: LCN

Retina
LCN

Connections: From-Retina

From-Retina

Threshold (%) 0 100

Scaling Factor [✓ 0.01] 0.001 0.0001 0.00001

Pre + Post + 0 100

Pre + Post - 0 100

Pre - Post + 0 100

Pre - Post - 0 100

HEBB RULE OFF [✓ ON]

Cancel

This menu allows the user to simulate models using a hebbian learning algorithm.

2.14.1 Buttons

1. **Cancel**
   Cancel the command and quit the menu.

2.14.2 Check Boxes

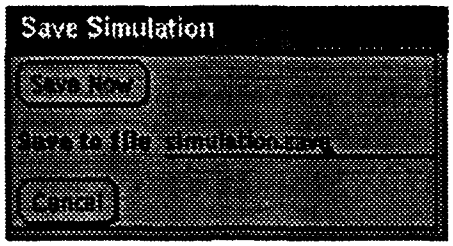
1. **HEBB RULE**
   When learning is "on", hebbian training is enabled. The user can toggle which specific connections are plastic by selecting the **Connections:** for the particular **Network Name:**. A connection which is surrounded by a box in the **Connections** scrolling list is plastic. Hebbian learning occurs when the networks are simulating, and weight changes occur using the following rule:

	Pre +	**Pre -**
Post +	↑	↓
Post -	↓	↓

A cell is considered (+) if its activity is above **Threshold(%)** of its maximum firing rate, otherwise it is (−). The magnitude of the weight change is controlled by the slider bars **Pre+ Post+**, **Pre+ Post-** ,**Pre- Post+** , **Pre- Post-**.

2. **Scaling Factor**
   Value to multiple the **Pre? Post?** slider bar values. For example, with **Scaling Factor** set to "0.001" and **Pre+ Post+** set at 50, the magnitude of the weight change for correlated pre/post activity would be .05.

2.15 Save Simulation



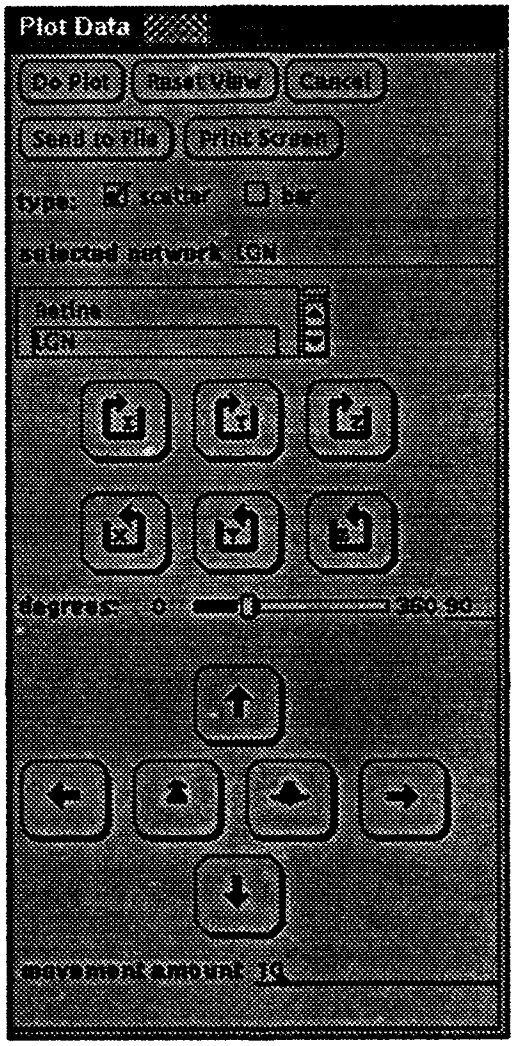
This menu allows the user to save the current state of the simulation.

2.15.1 Buttons

1. **Save Now**
   Saves all network, cells, connections and associated parameters currently loaded in the simulator to the file **Save to file**. The saved file is very large, since explicit connections are stored, and therefore users should be careful to limit their saved files if disk space is at a premium. A saved file can be reloaded into the simulator using the **Load Simulator** command on the NEXUS Control menu. Saved files load much quicker than NX files since connections are already made and do not need to be recalculated. The naming convention for a saved file is *filename*.save.

2. **Cancel**
   Cancel the command and quit the menu.

2.16 Plot Data



type: [✓ scatter] bar

selected network: LGN

degrees: 0 360 90

movement amount: 10

This menu allows the user to make 3D plots the activity levels of a selected network.

2.16.1 Buttons

1. **Do plot**
   Plot the activity of the **selected network**.

2. **Reset View**
   Reset view (axes) to original position.

3. **Send to File**
   Send the matrix of activity values for the **selected network** to the file **Output file**.

4. **Print Screen**
   Generate a Postscript file for the 3D plot in **NEXUS Plot Graphics** window.

5. **(rotate–arrows)**
   Rotate 3D plot around a particular axis by the amount **degrees**.

6. **(arrows)**
   Pan and zoom view for 3D plot by the amount **movement amount**.

7. **Cancel**
   Cancel the command and quit the menu.

2.16.2 Check Boxes

1. **type**
   Selects whether a scatter or bar graph of activity should be plotted.

3 Getting Started: An easy example

This section shows you how to get started using NEXUS with a simple example.[3]

A common pattern in biological sensory networks is the center-surround receptive field. The following example is a two layer network. The first layer represents an array of input activity, the second models the center-surround cells whose receptive fields are formed by a difference-of-gaussians connection pattern from the input layer.

3.1 NEXUS code

```
Network Input {
             # of units     = 576;
             x dimension    = 24;
             y dimension    = 24;
             transfer function = sigmoid(0.0,100.0,1.0);
             threshold      = const(0.0);
             decay          = 1.0;
             clamp          = on;
             initial firing rate = const(0.0);
             evaluation per cycle = 1;
}

Network Cent_Surr {
             # of units     = 400;
             x dimension    = 20;
             y dimension    = 20;
             transfer function = sigmoid(0.0,100.0,1.0);
             threshold= const(0.0);
             decay= 1.0;
             clamp= off;
             initial firing rate= const(0.0);
             evaluation per cycle= 1;
             connections {
                        from Input {
                             projection= full;
                             mapping type= direct;
                             connection field shape = ellipse;
                             length= 5;
                             width= 5;
                             rotation angle= 0;
                             shift x = 2;
                             shift y= 2;
                             weight function = dog(.3,.5,3.0,1.0);
                             feedback= off;
                        }
             }
}
```

[3]This example was contributed by Susan Courtney.

3.2 Using the simulator

1. Write the above code in emacs (or your favorite text editor) and save it in a file. It can be named anything, but for now let's refer to it as "first_network".

2. You are now ready to run NEXUS. Make sure you are in the same directory that your "first_network" file is in. Then type the command "nexus".

3. The first window is the settings window. For now, click on the *use defaults* button.

4. Now the control and display windows will appear. Place these wherever you like.

5. Click on the *Build Simulation* button on the control panel. Fill in the name of your code file, in this case "first_network", then click *OK*.

6. After the network is built, you may want to move and resize the networks in the display window. Click on *View*. Set the *selected network* to "Input" by clicking on the appropriate item in the scrolling list. Change the *scaling factor* to ".5". Resize the text displaying the name of the networks by clicking on *Text Dim (W)* and *Text Dim (H)*. Change the *scaling factor* to "2" and click *Net Dim* to change the display size of the *selected network*. Click on the *on* check box so that the *View Electrode* is on. Now you can move each network by selecting it with the left mouse button and placing it with the right mouse button (when the right mouse button is clicked, the top left edge of the network is place at the location of the pointer). The arrow buttons in the view window will pan left, right, up, down, and zoom in and out.

7. Now check the connections by clicking on *Display Connectivity*. Click the *on* check box for *Electrode* and the *on* check box for *Print Weights*. Now when you click on a cell in a network in the display window, the cells from which it receives input will be displayed in the color corresponding to the strength of the connection. Also, in the xterm window where you started "nexus", the connection strengths will be printed. If all the connections are what you expected, then proceed to the next step. If there is an error, or you wish to change the weight function, you can change the connection weights by clicking *Edit Connections*. Connections are edited by selecting a *Network name* and a *Connection* and then entering a new *Connection Function* using the NX format (such as "const(1.0)").

8. Click on *Display Activity*. The display should show all cells at zero activity.

9. Click on *Cell Activity*. Click on the *on* and *Change* heck boxes to enable the electrode to change cell activity. Then select a *Firing Rate* using the slider bar at the bottom of the window. You can now set the firing rate of a few of the input cells by clicking on those cells in the main display window. Try making an edge by setting all the cells in a line to some intermediate activity level.

10. Now click the *Simulate* button. Choose the *Sequential* check box, set *Swap firing rates* to *off*, and set the *Realtime Display* to *on*. Now...press *GO*. If the network is set up correctly and you made an edge on the input using the cell activity electrode, you should be able to see an edge enhancement or Mach Band effect in the activity of the center-surround network.

11. Click the *Save Simulation* button, fill in the file name, and press *Save Now*. Now, after you quit, when you run NEXUS again you will be able to load this network using the *Load simulation* button instead of having to re-build it.