Main Help Topics.

The following on-line help is available for Winneu:

Winneu basics. Description of program purpose and function.

Winneu operations. Description of possible actions and results.

Winneu Basics.

Purpose of Winneu.

Winneu is a program used for building, training and testing artificial neural systems. Building means constructing the structure of the net in terms of number of neuron-layers, number of neurons in each layer, type of neurons and neuron connections (synapses). Training means training the net for a certain behaviour, which is obtained by adapting neuron weights for each connection from a given set of training patterns (corresponding input and output neuron values). Testing means testing the net for fulfillment of the trained behaviour.

The purpose of Winneu is to be able to investigate different types of artificial neural nets. The intention is to present a tool which can be used to dissect a neural net down to its basic components. To do this a general implementation of the available net types is used. Each net type is therefore built of the same components using the same construction rules. The cost of using a unified approach is a relative slow performance. By using optimized implementations of each net a smaller learning and testing time could be obtained, but each net would then be isolated from the rest of the world. By using a unified approach an open architecture is obtained presenting an easy interface for both current use and future expansions. Though when using Winneu one should be aware that large nets may be time consuming.

The theoretical background for Winneu is obtained from the book: "Introduction to Artificial Neural Systems" by Jacek M. Zurada, West Publishing Company, ISBN 0-314-93391-3. For a short tutorial of neural nets see: neural net

This version of Winneu gives the possibility of handling the following neural net types:

Hopfield net
BAM net
single layer perceptron
multi layer perceptron

Because of the general implementation of the net types, it will be possible in future versions of Winneu to handle other net types, combine different net types and also make user-defined net types.

The world of Winneu.

Winneu consists of a main window with a menu bar and a number of child windows. The main window is the 'home' of a Winneu session and the child windows are Winneu tasks. The user interacts with the program via a command choice from the menu bar. A command choice may result in one or more dialog boxes to obtain further information for completing the command.

The three main data objects of Winneu are the defined nets, the net files and the pattern files. The defined nets are nets residing in computer memory (RAM) and ready for operation. The net files are saved nets not ready for operation. The pattern files contain data used by operating nets. A pattern file can be one of three types: train pattern file, test pattern file or result pattern file. The train pattern file contains data for training a net. The test pattern file contains data for testing a net. The result pattern file contains the result of testing a net.

The main operations (menu bar commands) of Winneu are divided into three groups according to the data object used. Each command group has a pull-down menu in the menu bar: the File menu for net file operations, the Net menu for defined net operations and the Pattern menu for pattern file operations. A pull-down menu called Suspend contains a command for suspending all activity in Winneu. Besides these four pull-down menus a menu called Training is appended when training a single layer perceptron or a multi layer perceptron. Also a pull-down menu, Window, exists when a child window is active. This menu provides commands for maintaining all child windows. See Operations for a description of all commands in Winneu.

Three types of child windows exist. First a child window for handling pattern files in a tabular form is used. When a table window is the active window extra commands are padded to the pull-down menu Pattern, conserning special commands for the pattern file opened. Next a child window type for displaying perceptron train graphics exists. Is the active window a train window, the pull-down menu Training is added to the menu bar containing special perceptron train commands. The last child window type is used for viewing perceptron train results. Mirrored 2-dim. decision lines of the trained n-dim. decision hyper-planes can be viewed together with the trained patterns.

Several instances of each child window type can be opened at the same time. One child window is the active and receives commands related to the type of window. The train windows opened are special because they all run simultaneously using a non-preemptive multitasking scheme. This means that perceptrons can be trained in the background while other tasks are dealt with both inside and outside Winneu.

Winneu Operations.

Operation categories.

Winneu operations are accessible from the menu bar and they are divided into six categories with a pull-down menu for each category. The main commands are located in the three menus File, Net and Pattern according to the three data objects used in Winneu: the net file, the defined net and the pattern file. The last three menus are the Suspend menu, the Training menu and the Window menu. All operations in each category are explained in the following.

File menu operations.

The File menu contains commands used on net files residing on disc. A net file is used to store a defined net for later use. All net files are expected to be in the subdirectory NET of the directory from where Winneu was started. The File menu commands are:

Load: Load a saved net file and define a new net.
Copy: Copy a saved net file.
Save: Save a defined net - both trained and untrained nets can be saved.
Delete: Delete a net file from disc.
Exit: Exit program. All child windows containing unsaved data and all unsaved defined nets are prompted for saving before exit.

Net menu operations.

The Net menu operations are used to deal with defined nets residing in computer memory (RAM). A defined net is a net ready for operation using data stored in pattern files. The Net menu commands are:

Build: Build and define a new net.
Train: Train a defined net from a train pattern file in the subdirectory PATTERN with the same name as the net. Already trained nets can be trained again on top of old results.
Test: Test a defined net from a test pattern file in the subdirectory INPUT with the same name as the net. The results of the testing is saved in a result pattern file in the subdirectory OUTPUT also with the same name as the net.
View: View net structure or train results of a defined net. Only perceptron train results can be viewed.
Delete: Delete a defined net from computer memory (RAM).

Pattern menu operations.

Operations in the pattern menu handles pattern files. Pattern files contain data used by defined nets. Three kinds of pattern files exist: train pattern files in subdirectory PATTERN, test pattern files in subdirectory INPUT and result pattern files in subdirectory OUTPUT. The Pattern menu contains two kinds of commands separated by a horizontal line. The upper commands relates to pattern files not opened and the lower commands are used on the active opened pattern file. The lower commands are padded to the menu when a table child window is the active window. The upper Pattern menu commands are:

New: Create a new pattern file.
Copy: Copy a pattern file.

Edit: Edit a pattern file. Only train pattern files and test pattern files can be edited. It is considered to be of no use to edit a result pattern file (Do not manipulate test results!!!).
View: View contents of a pattern file. Here the result pattern files can be viewed.
Delete: Delete a pattern file.

The lower Pattern commands are:

Save: Save current active opened pattern file if any changes have been made.
Close: Close current active opened pattern file. Prompt for saving if changes have been made.

Suspend menu operations.

Suspend: Command used to suspend all activity in Winneu, or to abort a time consuming command which have suspended Winneu.

Training menu operations.
The Training menu is added to the menu bar when a perceptron training window is the active window. This menu contains commands used to influence the train session of a single layer perceptron or a multi layer perceptron. The Training menu commands are:

Suspend: Suspend or un-suspend training in current train window.
Stop: Stop training in current train window and close window.
Settings
- **Error rate:** Change the accepted error rate.
- **Learning rate:** Change the learning rate.

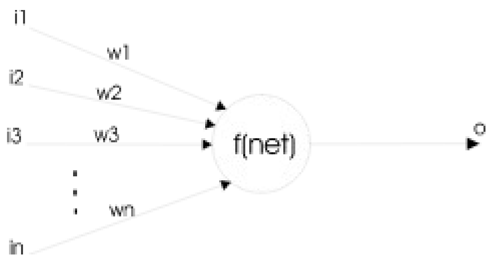
Window menu operations.
The Window menu commands are used for maintaining all child windows opened. A Winneu child window is an object in the Winneu environment and can be displayed in different ways. By using Window menu commands the child windows can be organized and displayed in different ways. The Window menu commands are:

Cascade: Cascade opened child windows not minimized. Child windows are placed on top of each other with a piece of each window visible.
Tile: Tile child windows. Child windows are put next to each other filling the parent window.
Arrange Icons: All minimized child windows are arranged.
Close Active: The active child window is closed after evt. prompting for saving changed data.
Close All: All child windows are closed. Any child window containing unsaved data are prompted for saving.

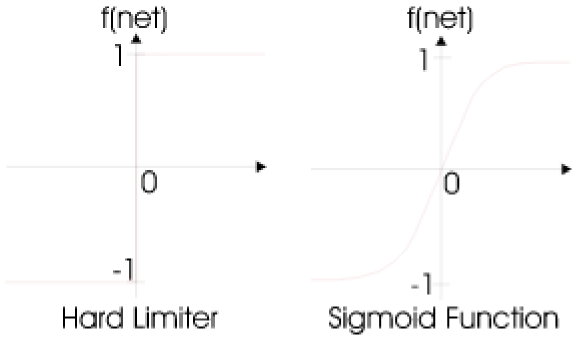
Tutorial of neural nets.

The neuron.

A neural net (e.g. the human brain) is a collection of neurons connected to each other. In Winneu the following simplified model of a neuron is used:

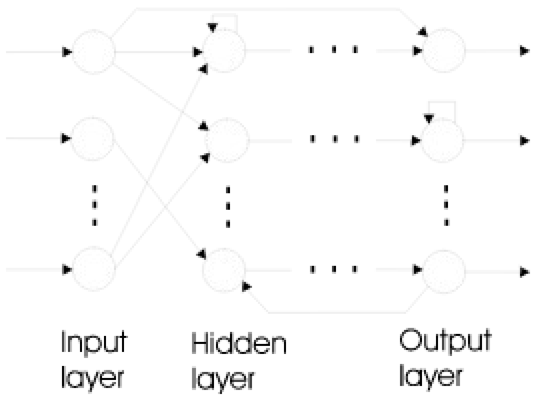


The neuron model (from now on referred to as the neuron) consists of some weighted incomming signals (dendrites), a neuron activation function (cell body) and an outgoing signal (axon). The outgoing signal is determined as the activation function used on the sum of the weighted incomming signals. The last incomming signal is always -1 and the weight T is interpreted as a threshold value. The sum of the other weighted incomming signals has to exceed T before the neuron is said to trigger (fire). A neuron (neural net) is said to be **unipolar** when all signals are between 0 and 1, and **bipolar** when all signals are between -1 and 1. A neurons inputs and output can either be **continuous** (continuous values) or **binary** (binary values). The activation function types used in Winneu are the **hard limiter** and the **sigmoid function**. The bipolar version of these two functions are:



The neural net.

The neurons in a neural net can be connected in a variety of ways. In Winneu the neurons are ordered in layers. The three types of layers used are the **input layer**, the **hidden layer** and the **output layer**. The input layer receives input which are to be computed, the hidden layer(s) contain neurons not connected to the outside and the output layer contains the result of the computation. A net has to have an input layer and an output layer, but the hidden layer may not exist. The following figure shows the general layout of a Winneu neural net:



Training of the neural net.

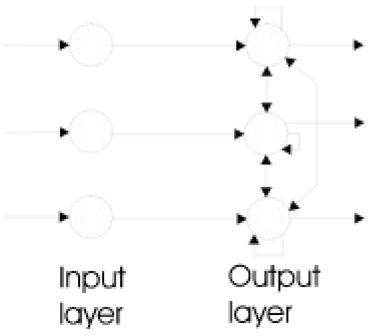
A neural net is trained from a set of **train patterns**. Each train pattern consists of an input value for each input neuron and of an output value for each output neuron. The purpose of the training is to teach the net to respond with the correct output pattern when presented to an input pattern. The training is accomplished by setting the weights of the neurons. In Winneu a neural net can be trained either by setting the weights directly or adaptively. The Hopfield net and the BAM net are trained directly and the single layer perceptron and the multi layer perceptron are trained adaptively.

Testing of the neural net.

When testing a neural net a set of **input patterns** are presented to the net and the corresponding **output patterns** are calculated. In Winneu both untrained and trained nets can be tested. The results of the testing depend on the type of net.

Hopfield net.

Ex. of layout.



Defining a Hopfield net in Winneu.

When defining a Hopfield net in Winneu only the number of input neurons has to be specified, because the net has the same number of input and output neurons, has no hidden layers and is bipolar, binary by definition. The activation function used is the hard limiter.

Description.

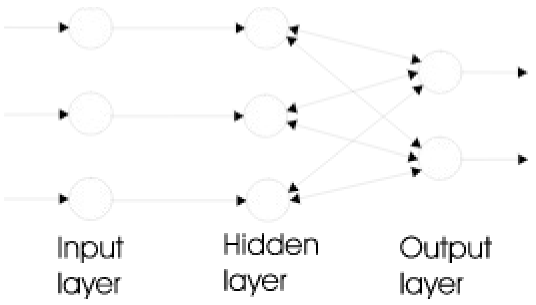
The Hopfield net is a recurrent autoassociative memory used for storing and restoring binary patterns. When a distorted pattern is presented to a trained Hopfield net a resulting output pattern of the same dimension as the input pattern is found. The output pattern found is the trained pattern which resembles the distorted pattern the most. The net is recurrent meaning the net contains feedback, so when using the net an input pattern is set and the net runs dynamically until a stable output pattern (attractor, equilibrium state) is obtained. The input type to the Hopfield net is bipolar, binary which means values -1 and 1 can be used, but when testing a net the input may also be 0 indicating a not known value. If the right output pattern is found all 0's are set to the right values.

Training.

The Hopfield net is trained from information known a priori, which means the binary train patterns directly form the weight matrix. When training the net the weight matrix is calculated from the train patterns and the weights are filled in the net strucure. A trained net can be further trained by simply adding a new weight matrix to the already trained net. The train pattern values must be either -1 or 1.

BAM net.

Ex. of layout.



Defining a BAM net in Winneu.

When defining a BAM net in Winneu only the numbers of input and output neurons have to be specified, because the net is bipolar, binary by definition. One hidden layer is used with the same number of neurons as the input layer. The activation function used is the hard limiter.

Description.

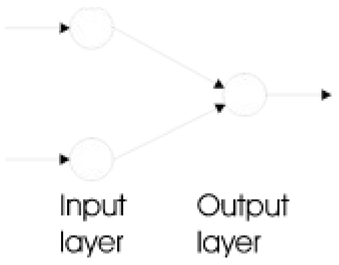
The BAM (Bidirectional Associative Memory) net is a recurrent heteroassociative, content-addressable memory used for storing and restoring binary pattern pairs. When a distorted pattern is presented to a trained BAM net, the net finds the stored input pattern which resembles the most. The output of the net is then set to the output pattern stored together with the found input. In other words the net makes an association from the input towards something different from the input - it's something like thinking of hell when seeing your mother-in-law. The net is recurrent meaning the net contains feedback, so when using the net an input pattern is set and the net runs dynamically until a stable output-input pattern pair (attractor, equilibrium state) is obtained. The input type to the BAM net is bipolar, binary which means values -1 and 1 can be used, but when testing a net the input may also be 0 indicating a not known value. If the right output pattern is found all 0's are set to the right values.

Training.

The BAM net is trained from information known a priori, which means the binary train pattern pairs directly form the weight matrix. When training the net the weight matrix is calculated from the train patterns and the weights are filled in the net strucure. A trained net can be further trained by simply adding a new weight matrix to the already trained net. The train pattern values must be either -1 or 1.

Single Layer Perceptron.

Ex. of layout.

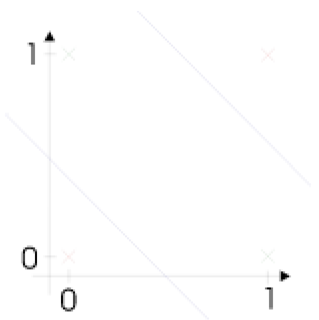


Defining a single layer perceptron in Winneu.

When defining a single layer perceptron the numbers of input neurons and output neurons have to be specified. Also the type of net (unipolar or bipolar) and the output type (binary or continuous) have to be determined. The number of hidden layers are 0 and the input type is continuous. The activation function used is the sigmoid function.

Description.

The single layer perceptron is a trainable decision-making feedforward network used to determine class membership of an input pattern. Feedforward means that there is no feedback in the net, so all signal flow goes from input to output. When testing the net output will be ready at a known time after inputs are set. The single layer perceptron can determine the nearest trained class in the n-dimensional space for a pattern of dimension n. Two classes (categories) can be trained for one output neuron. These two classes determine two discriminant functions used for determining the class membership of a random input pattern. A pattern is said to belong to a class, when the discriminant function of this class yields the largest value computed from the pattern values. The two discriminant functions result in one decision surface (hyperplane of dimension n-1) which indicate the border between the two classes. The decision surface is points in the n-dim. space where the discriminant functions yields the same values. The single layer perceptron can only produce one decision surface of dim. n-1 for each output neuron because only input neurons and output neurons exists. This is why the problem of the Exclusive-Or never can be solved by a single layer perceptron. The EX-OR needs two decision surfaces (lines) as shown in the following figure:



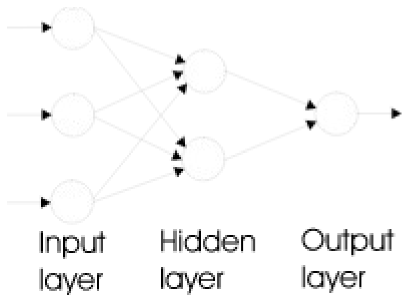
The EX-OR problem can be solved by the multi layer perceptron because this net has one or more hidden layers of neurons applying more decision surfaces. The reason for dividing perceptrons in single layered and multi layered in Winneu is, that the single layer perceptron can both have binary and continuous outputs while the multi layer perceptron can only have continuous outputs due to the back-propagation error training scheme.

Training.

The single layer perceptron is trained using two different schemes - one for binary outputs and one for continuous outputs. The continuous output training scheme is the back-propagation error scheme used by the multi layer perceptron. Here training is accomplished by applying the same train patterns to the net and update the neuron weights until a desired error rate is reached. When training the binary output version the updating of the weights are slightly different and the algorithm is continued until the error is 0.

Multi Layer Perceptron.

Ex. of layout.



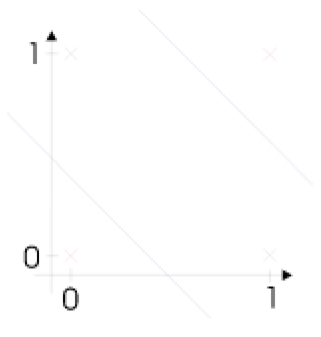
Defining a multi layer perceptron in Winneu.

When defining a multi layer perceptron the numbers of input neurons, hidden layers, neurons in hidden layers and output neurons have to be specified. Also the type of net (unipolar or bipolar) has to be determined. The input and output type is continuous. The activation function used is the sigmoid function.

Description.

The multi layer perceptron is a trainable decision-making feedforward network used to determine class membership of an input pattern. Feedforward means that there is no feedback in the net, so all signal flow goes from input to output. When testing the net output will be ready at a known time after inputs are set. The multi layer perceptron can determine the nearest trained class in the n-dimensional space for a pattern of dimension n. The number of classes (categories) trained, R, determines R discriminant functions used for determining the class membership of a random input pattern. A pattern is said to belong to a class, when the discriminant function of this class yields the largest value computed from the pattern values. The discriminant functions result in decision surfaces (hyperplanes of dimension n-1) which indicate the borders between classes. The decision surfaces are points in the n-dim. space where two or more dominating discriminant functions yields the same values.

The problem of the Exclusive-Or can be solved by a multi layer perceptron defined minimum as having two input neurons, one hidden layer with two neurons and one output neuron. The demand of minimum two decision surfaces (lines) for solving the EX-OR problem is fulfilled for this net. The decision lines for the trained EX-OR net is shown in the following figure:



The reason for dividing perceptrons in single layered and multi layered in Winneu is that the single layer perceptron can both have binary and continuous outputs while the multi layer perceptron can only have continuous outputs due to the back-propagation error training scheme.

The force of the multi layer perceptron is its ability to generalize from a complex input. The net can be viewed as dividing its world (the n-dim. input space) into a definit number of boxes (classes, categories), and whenever an input pattern is applied, the box which is triggered (found) by the net is delivered as the output. When Sherlock Holmes postulates that he deductively finds the murderer by combining facts logically, he is wrong. What happens is that the facts act as the input pattern to the well-trained mind of Sherlock Holmes and the profile of the murderer is triggered by these facts. Using this profile he usually finds the guilty man (woman) in the surroundings. As a curiosity one can say, that the reader of Sir Arthur C. Doyles books uses the tecnique described by the BAM net to make associations between the words in a book and the world of Sherlock Holmes.

If a solution exists to a given classification problem finding the optimal solution is determining the best net structure and the best train patterns for this structure. How to do this? - well, if I knew I wouldn't sit here cursing my slow 386. Some rules of thumb for determining the net structure do exist though. The numbers of input and output neurons are defined by the problem, but the number of hidden layers and the number of neurons in each hidden layer are free design parameters. It can be proved, that any continuous function can be estimated by a multi layer perceptron with one hidden layer, but in some cases using two hidden layers may require a smaller number of connections. It may be an advantage using more than one hidden layer for very complex problems. The choice of hidden neurons can be made from different experiments. Either try different numbers, start with a small number and then increase or start with a high number and then decrease. The experiment with the best results is then chosen.

Training.

The multi layer perceptron is trained using the back-propagation error scheme. Here training is accomplished by applying the same train patterns to the net and update the neuron weights until a desired error rate is reached. The choice of train patterns is essentail for obtaining a good training. If the patterns cover the essential classes (categories) too sparsely the net may not be able to find the wanted minimum of the error (error rate) but may stabilize on a higher value, or the net may generalize too roughly throwing away valuable information. If the patterns are too detailed the training may result in the wanted error rate, but it may take very long time and when finished the net will not be able to generalize (peel off unnecessary information).

Build Net Data dialog box.

Choice of net name and net type used for building a new net.

Name of Net

Must start with character a-z. All higer case characters are converted to lower case. Max. name length is 8 characters.

Type of Net

Hopfield net
BAM net
single layer perceptron
multi layer perceptron

Net Data for Hopfield Net dialog box.

Net data for Hopfield net used for building a new net.

Number of neurons

Number of neurons in input layer and output layer. Min. value is 1 and max. value is 999999.

Net Data for BAM Net dialog box.

Net data for BAM net used for building a new net.

Number of input neurons

Number of neurons in input layer. Min. value is 1 and max. value is 999999.

Number of output neurons

Number of neurons in output layer. Min. value is 1 and max. value is 999999.

Net Data for Single Layer Perceptron dialog box.

Net data for single layer perceptron used for building a new net.

Number of input neurons

Number of neurons in input layer. Min. value is 1 and max. value is 999999.

Number of output neurons

Number of neurons in output layer. Min. value is 1 and max. value is 999999.

Activation function type

Range of signal values in net.
unipolar: Values are between 0 and 1.
bipolar: Values are between -1 and 1.

Output neuron type

discrete: Output values are either 0 or 1 for unipolar nets and -1 or 1 for bipolar nets.
continuous: Output values can be any value in the range of signal values determined by the activation function type.

Net Data for Multi Layer Perceptron dialog box.

Net data for multi layer perceptron used for building a new net.

Number of input neurons

Number of neurons in input layer. Min. value is 1 and max. value is 999999.

Number of output neurons

Number of neurons in output layer. Min. value is 1 and max. value is 999999.

Number of hidden layers

Number of hidden layers between input layer and output layer. Min. value is 1 and max. value is 999.

Neurons: Number of hidden neurons for each hidden layer. Min. value is 1 and max. value is 999999. A value for each hidden layer has to be entered. Hidden layers are numbered starting at the end of the input layer and ending at the beginning of the output layer.

Activation function type

Range of signal values in net.

unipolar: Values are between 0 and 1.

bipolar: Values are between -1 and 1.

Train Session Data for Perceptron dialog box.

Train graphics data for a single layer perceptron and a multi layer perceptron. The train plot of a train session consists of none or up to four plots (cells).

Error rate

Accepted error rate determining when the train session is finished. Min. value is 0, max. value is 999.999 and default value is 0.1.

Learning rate

The learning rate determines the strength of the updating of all weights. Min. value is 0, max. value is 999.999 and default value is 0.5.

Plot cell data

Cell 1: Cell plot data for cell 1.
Cell 2: Cell plot data for cell 2.
Cell 3: Cell plot data for cell 3.
Cell 4: Cell plot data for cell 4.
No train graphics: Indicate no train graphics.

Train Plot Cell Data dialog box.

Data for one plot cell of perceptron train graphics.

Cell type for cell x

Type of graphics in cell x (x = 1, 2, 3 or 4).

Error plot 1: Plot of training error. The plot is cleared and old data lost when right end of plot reached. The update rate is constant and determined by the user before start of training (see below). Only one error plot 1 is allowed.

Error plot 2: Plot of training error. Old plot data is compressed and the plot is updated when right end of plot reached. The update rate is initially 1 and is doubled after each update. When the training is completed error plot 2 shows the entire development in the error, while error plot 1 only shows one screen back in time. Only one error plot 2 is allowed.

Line plot: Plot of decision lines and evt. train patterns. The line plot shows the development of the n-dimensinal decision hyper-planes mirrored in the 2-dimensional plane. This means that for a target neuron the decision plane according to two incomming neurons can be displayed. Only planes for incomming neurons in the same layer can be viewed in the same line plot. The update rate is constant and determined by the user before start of training. Max. two line plots are allowed.

Update rate

Determines how often a plot is updated. A value of x means that the plot is updated after x trainings with all train patterns. Min. value is 1 and max. value is 999999. The update rate can be applied for a training plot 1 (default value 1) and line plots (default value 5).

Layer data

Only for line plots. Determines from which layer the incomming neurons of a decision line is chosen.

Input layer: Incomming neurons are chosen from the input layer.

Plot patterns: Indicate plot of train patterns. If patterns are to be plotted two input neurons and one output neuron have to be chosen indicating which part of the patterns to plot.

Hidden layer: Incomming neurons are chosen from a hidden layer. Plot of train patterns not possible.

Hidden layer nr: Number of hidden layer for incomming neurons.

Number of lines

Only for line plots. Indicate number of decision lines to plot. Min. value is 1 max. value is 100.

Line data: Enter for each decision line the numbers of the two incomming neurons and the number of the target neuron.

View Net dialog box.
Choose net to view and type of viewing.

Nets defined
List of nets currently defined and possible to choose to view.

Type of net
Net type of net chosen from list of defined nets.

Number of input neurons
Number of neurons in input layer of net chosen from list of defined nets.

Number of output neurons
Number of neurons in output layer of net chosen from list of defined nets.

View
Type of viewing.
Net structure: View structure of net.
Train results: View train results of net. Only available for trained single layer perceptrons and multi layer perceptrons.

Net Structure Data dialog box.

Presentation of structure data for defined net.

Net name

Name of net.

Net type

Type of net. Possible net types:

- Hopfield net
- BAM net
- single layer perceptron
- multi layer perceptron

Act. function type

Type of neuron signals. Either unipolar (between 0 and 1) or bipolar (between -1 and 1).

Train state

Current state of training. Either trained or untrained.

Neuron data

Input neurons: Number of neurons in input layer and type of neurons (binary or continuous).
Hidden layers: Number of hidden layers between input layer and output layer.
Output neurons: Number of neurons in output layer and type of neurons (binary or continuous).
Layer nr.: Choice of hidden layer to investigate.

- **Hidden neurons:** Number of hidden neurons in hidden layer chosen, if valid layer number chosen.

Type of Pattern File dialog box

Choice of pattern type and file type for a new pattern file.

Type of patterns

Train patterns: Patterns for training a net.
Test patterns: Patterns for testing a net.

Type of file

Owned by net: Indicate if the pattern file is owned by a defined net. The defined net later chosen then determines the structure of the patterns. If the pattern file is not owned by a net the structure of the patterns has to be explicitly given by the user.

Data for Train Pattern File dialog box.

Defining the structure of a new train pattern file.

Name of pattern file

Must start with character a-z. All higer case characters are converted to lower case. Max. name length is 8 characters.

Activation function type

Range of signal values in net.
unipolar: Values are between 0 and 1.
bipolar: Values are between -1 and 1.

Number of input neurons

Number of input values for each pattern. Min. value is 0 and max. value is 999999. The minimum value is 0 because a Hopfield net only has output train patterns.

Input neuron type

discrete: Input values are either 0 or 1 for unipolar nets and -1 or 1 for bipolar nets.
continuous: Input values can be any value in the range of signal values determined by the activation function type.

Number of output neurons

Number of output values for each pattern. Min. value is 1 and max. value is 999999.

Output neuron type

discrete: Output values are either 0 or 1 for unipolar nets and -1 or 1 for bipolar nets.
continuous: Output values can be any value in the range of signal values determined by the activation function type.

Data for Test Pattern File dialog box.

Defining the structure of a new test pattern file.

Name of pattern file

Must start with character a-z. All higer case characters are converted to lower case. Max. name length is 8 characters.

Activation function type

Range of signal values in net.
unipolar: Values are between 0 and 1.
bipolar: Values are between -1 and 1.

Number of input neurons

Number of input values for each pattern. Min. value is 1 and max. value is 999999.

Input neuron type

discrete: Input values are either 0 or 1 for unipolar nets and -1 or 1 for bipolar nets.
continuous: Input values can be any value in the range of signal values determined by the activation function type.

Train Graphics Data dialog box.

Train graphics view data for a single layer perceptron or a multi layer perceptron. The graphics plot of the train results of a trained perceptron consists of 1, 2, 3 or 4 plots (cells).

Plot cell data

- **Cell 1:** Graphics plot data for cell 1.
- **Cell 2:** Graphics plot data for cell 2.
- **Cell 3:** Graphics plot data for cell 3.
- **Cell 4:** Graphics plot data for cell 4.

Train Graphics Cell Data dialog box.

Data for one plot cell of train results graphics.

Layer data for cell x (x=1, 2, 3, or 4)

Determines from which layer the incomming neurons of a decision line is chosen.

Input layer: Incomming neurons are chosen from the input layer.

Plot patterns: Indicate plot of train patterns. If patterns are to be plotted two input neurons and one output neuron have to be chosen indicating which part of the patterns to plot.

Hidden layer: Incomming neurons are chosen from a hidden layer. Plot of train patterns not possible.

Hidden layer nr: Number of hidden layer for incomming neurons.

Number of lines

Indicate number of decision lines to plot. Min. value is 1 max. value is 100.

Line data: Enter for each decision line the numbers of the two incomming neurons and the number of the target neuron.