

Chapter 8

Neuralyst Operations Reference

8.1 Neural Network Configuration Menu

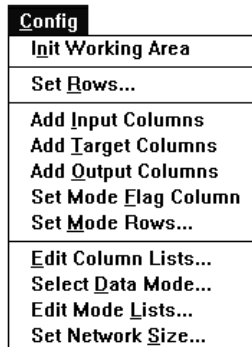


Figure 8-1 Config Menu

The **Config** menu provides you with access to the commands necessary to define the structure of a problem and the associated neural network to Neuralyst. The commands are ordered in the same sequence that you would generally proceed in defining the problem to Neuralyst, though this is not a requirement in all situations (in particular the various Add/Edit Column commands may be used in any order).

8.1.1 Init Working Area

Function: **Init Working Area** is used to reserve the area that Neuralyst will use for its operations.

Usage: To use **Init Working Area**, select the cell that is at the upper left corner of the area to be reserved and execute the command. All cells to the right of the specified cell and all cells below the specified cell will be included in the Working Area. It is usually convenient to designate the Working Area starting with the first cell at the top of the sheet and to the right of other data.

Effect: **Init Working Area** will first confirm the intent to overwrite the region. If this is confirmed it will clear the region. Starting from a clear region, it will build its Working Area by placing the Title Block, Statistics Block, Parameter Block, Row Description Block, Column Description Block, Network Description Block, and Network Weights Block in this region (see Section 8.3 for a description of these data blocks). These blocks will be initialized to default values.

Warnings: **Init Working Area** must be executed for a new sheet before any other configuration operations can occur. Any data that is in the Working Area prior to the execution of the command will be deleted; therefore Neuralyst will always ask for confirmation prior to executing the command. Any data that is placed in the Working Area after the execution of the command may cause incorrect operation of Neuralyst or be changed by Neuralyst depending on the actual position within the Working Area. Executing **Init Working Area** a second time in the same or different location will result in the old area being ignored and the new area being set up and used.

8.1.2 Set Rows

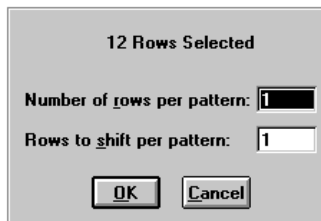


Figure 8-2 Set Rows Dialog Box

Function: **Set Rows** is used to designate the rows that will be input to the neural network by Neuralyst.

Usage: To use **Set Rows**, select the rows that will be input and execute the command. After the region is set, a dialog box will show the number of rows contained in the selected region and will then request the number of rows per pattern (other appropriate terms might be instance or example) and the number of rows to shift per pattern.

When a worksheet is modified by inserting or deleting one or more rows so that the range of rows is no longer the same as that previously configured for Neuralyst, then a **Set Rows** on the new range should be performed.

Effect: The first and last rows in the selected region will be retained as the lower and upper bounds, respectively, for training, testing, or running patterns with the neural network. For example, when Rows per Pattern is set to 3, (while Rows to Shift is retained at its default value of 1) the window is three rows high. The window will step down one row for each pattern, this means that there is a two row overlap with each previous pattern.

To change the overlap, Rows to Shift may be changed. Increasing it to 2 in the example would mean the window would step down two rows for each pattern, creating a one row overlap with each previous pattern. Increasing it to 3 in the example, would mean the window would step down three rows for each pattern, resulting in no overlap between patterns. Setting the Rows per Pattern and Rows to Shift to equal values is how two-dimensional input patterns are defined.

Warnings: It will not matter which column or columns are in the selected region used for **Set Rows**; only the row numbers will be used. Only one contiguous region may be selected; discontinuous extended regions will not be handled properly. Selection and execution of **Set Rows** on a second region will change the row numbers to those of the new region.

There may be blank rows in the selected region; blank rows will always be ignored and skipped. Note that this includes the operation of Rows per Pattern and Rows to Shift per Pattern; blank rows will not be counted in these either.

In addition, partially blank rows or rows with non-numeric data at the beginning or end of the selected region will be skipped. Data in

partially blank rows or rows with non-numeric data that are interspersed with valid rows in the selected region will be treated as 0's in the case of blanks or cause an error in the case of non-numeric data.

8.1.3 Add Input Columns

Function: **Add Input Columns** is used to designate those columns that will be used as inputs to the neural network for training, testing, or running patterns.

Usage: To use **Add Input Columns**, select one or more columns that contain components of the input pattern and execute the command. **Add Input Columns** may be repeated; each repetition will add those columns selected to the list of previously selected columns.

Effect: The columns selected will be noted and used as components of the input patterns to the neural network. Only those cells of the columns selected in the range established by **Set Rows** will be used.

Warnings: It will not matter which rows in the column are used to select the column; only the columns themselves will be set. The rows used will be determined by the selection passed to **Set Rows**; each column will use the same rows. All cells in the region to be used for input patterns must contain valid numeric or symbolic values; if there are any invalid values (other than blank rows) in the region determined by the intersection of rows as set by **Set Rows** and columns as set by **Add Input Columns**, then there will be an error.

8.1.4 Add Target Columns

Function: **Add Target Columns** is used to designate those columns that will be used as targets (goals or known outputs) to the neural network for training or testing input patterns.

Usage: To use **Add Target Columns**, select one or more columns that contain components of the known output pattern and execute the command. **Add Target Columns** may be repeated; each repetition will add those columns selected to the list of previously selected columns.

Effect: The columns selected will be noted and used as components of the target patterns for the neural network to compare with its own outputs. Only those cells of the columns selected in the range established by **Set Rows** will be used.

Warnings: It will not matter which rows in the column are used to select the column; only the columns themselves will be set. The rows used will be determined by the selection passed to **Set Rows**; each column will use the same rows. All cells in the region to be used for target patterns must contain valid numeric or symbolic values; if there are any invalid values (other than blank rows) in the region determined by the intersection of rows as set by **Set Rows** and columns as set by **Add Target Columns**, then there will be an error. The number of Target columns should match the number of Output columns, otherwise there could be an error.

8.1.5 Add Output Columns

Function: **Add Output Columns** is used to designate those columns that will be used as outputs for the neural network after training, testing, or running input patterns.

Usage: To use **Add Output Columns**, select one or more columns that are available to hold the output pattern and execute the command. **Add Output Columns** may be repeated; each repetition will add those columns selected to the list of previously selected columns.

Effect: The columns selected will be noted and used as components of the output patterns from the neural network after it processes the input patterns. Only those cells of the columns selected in the range established by **Set Rows** will be used.

Warnings: It will not matter which rows in the column are used to select the column; only the columns themselves will be set. The rows used will be determined by the selection passed to **Set Rows**; each column will use the same rows. All cells in the region to be used for output patterns will be overwritten by the neural network outputs; if there are any values in the region determined by the intersection of rows as set by **Set Rows** and columns as set by **Add Output Columns**, then these will be overwritten. The number

of Target columns should match the number of Output columns, otherwise there could be an error.

8.1.6 Set Mode Flag Column

Function: **Set Mode Flag Column** is used to designate the column whose values will be used to switch Neuralyst from training mode to testing or running mode on a pattern by pattern basis. The column also holds special Neuralyst Mode Flags to indicate a MIN, MAX, or SYMBOL, rows.

Usage: To use **Set Mode Flag Column**, select the column that contains the flags designating whether the pattern on that row is to be used as a training pattern, as a testing/running pattern, or special Mode Flags, and execute the command.

Effect: The column selected will be noted and for each pattern window, as determined by the rows selected by **Set Rows** and the pattern window parameter, Rows per Pattern, Neuralyst will check the value of the cell in the Mode Flag column on the same row as the last row of the pattern window. If the value is TRAIN the pattern will be used for training and if the value is TEST the pattern will be used for testing or running. If the value is blank, then it is treated as TRAIN. Testing is distinguished from running only by the presence of values on the relevant rows of the Target columns. If the value is MIN, MAX, or SYMBOL, the pattern will be skipped for training, testing or running purposes; but will be interpreted appropriately otherwise.

Warnings: It will not matter which rows in the column are used to select the column; only the column itself will be set. The rows used will be determined by the selection passed to **Set Rows**. Only the text values, TRAIN, TEST, MIN, MAX, SYMBOL, or blank cells should be in the Mode Flag Column; other values may cause an error. Only one Mode Flag Column may be set; if another column is selected and **Set Mode Flag Column** is executed, then the new column will replace the old column. If a Mode Flag Column is not set then both **Train Network**, **Run/Predict with Network**, and **Run Genetic Supervisor**, will use all the defined rows. Only one instance each of MIN, MAX and SYMBOL may be present in the Mode Flag column. Additional instances may cause errors.

8.1.7 Set Mode Rows

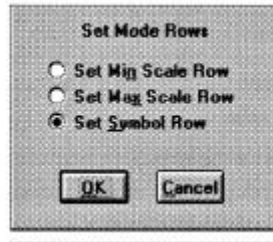


Figure 8-3 Set Mode Rows Dialog Box

Function: **Set Mode Rows** is used to designate and optionally initialize the rows which will be identified with special Mode Flags which will indicate to Neuralyst that those rows have special functions.

Usage: To use **Set Mode Rows**, select the row that contains the Mode Flags designating whether the data on that row is to be used as a MIN, MAX, or SYMBOL, row and execute the command.

Effect: The row selected will be identified with a Mode Flag of MIN, MAX, or SYMBOL, in accordance with the type selected from the dialog box. Neuralyst will then treat the information set in those rows in accordance with the Mode Flag set. Each column of a MIN row can be initialized at the time of the command automatically with the scanned minimum numeric values, or later by using **Edit Mode Lists** with scanned or user defined numeric or symbolic values, which will be taken by Neuralyst to be the minimum value defined value for that row. Each column of a MAX row can be initialized at the time of the command automatically with the scanned maximum numeric values, or later by using **Edit Mode Lists** with scanned or user defined numeric or symbolic values, which will be taken by Neuralyst to be the maximum defined value for that row, subject to additional headroom added by Scaling Margin. Each column of a SYMBOL row can be filled in later by using **Edit Mode Lists** with a Symbol List defining the valid Symbols for that row.

Warnings: It will not matter which columns in the row are used to select the row; the row will be set with the selected Mode Flag and the column used to identify the row will be determined by the column

selection passed to **Set Mode Flag Column**. Only one row can be selected of each Mode Flag type. If a mode row has been previously set of the type selected in the dialog box, then **Set Mode Rows** will overwrite the previous Mode Flag. The command will automatically initialize the Min or Max fields with the respective Min or Max numeric values from the currently defined Input or Target columns, if feasible and confirmed. Only the Mode Flag values, MIN, MAX, or SYMBOL, will be entered by **Set Mode Rows**. The only other Mode Flag values that are legal for the Mode Flag Column are TRAIN, TEST or blank; other values may cause an error. Before executing a **Set Mode Rows** command, a Mode Flag Column must have been set first with **Set Mode Flag Column**.

8.1.8 Edit Column Lists

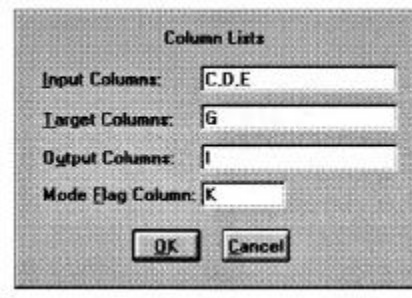


Figure 8-4 Edit Column Lists Dialog Box

Function: **Edit Column Lists** allows you to make additions or deletions to existing settings for Input, Target, Output and Mode Flag columns. Note that this is the only way to delete columns from the lists.

Usage: To use **Edit Column Lists** execute the command. A dialog box will appear listing the columns of each type. These lists may be edited with mouse and keyboard operations as in any Windows data entry box. The format for column lists is a series of one or two letter column names separated by spaces or commas. When complete you confirm **OK** to accept the changes or **Cancel** to abort the changes.

When a worksheet is modified by inserting or deleting a column so that some of the columns previously configured for Neuralyst no longer have the same column names, then **Edit Column Lists** may

be used to revise the column lists to reflect these changes. Without **Edit Column Lists**, it would be necessary to perform an **Init Working Area** and rebuild the configuration.

Effect: When the command is executed, Neuralyst will put up the Edit Column Lists Dialog Box listing the column information entered to that point by previous **Add Column** commands. Adding new columns to any list will not be treated any differently than if the column was added by an **Add Column** command. Deleting a column already on the list will cause Neuralyst to “forget” that the column was ever entered. Once you have completed editing and confirmed the changes with **OK**, Neuralyst will enter the revised data into its column lists.

Warnings: All considerations listed for the various **Add Column** commands apply for this method of column deletion or addition. In addition, changing the column lists, other than the Mode Flag Column, will require the **Set Network Size** command to be re-executed for the new set of columns, thus forgetting any learning that has been done. In this case Neuralyst will first ask you to confirm your intentions.

8.1.9 Select Data Mode

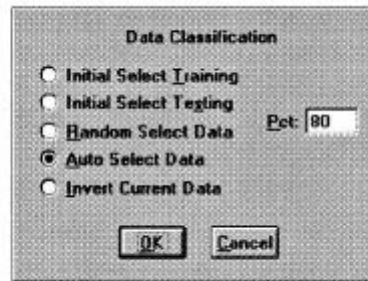


Figure 8-5 Data Classification Dialog Box

Function: Select Data Mode is used to automate the classification of data between training data and testing data.

Usage: To use **Select Data Mode**, make sure that all rows and columns are first defined, then execute the command. A dialog box will appear asking for one of five options, Initial Select Training,

Initial Select Testing, Random Select Data, Auto Select Data, and Invert Current Data. Select the desired option and confirm with **OK**. The operation can be aborted with **Cancel**.

Effect: When one of the five options: Initial Select Training, Initial Select Testing, Random Select Data, Auto Select Data, and Invert Current Data, is selected, the Mode Flag column for each row associated with a pattern window will be set to TRAIN or TEST, corresponding to the classification of the data in the pattern window associated with that row as training data or testing data. A percent setting is associated with Initial Select Training, Initial Select Testing, Random Select Data, and Auto Select Data.

Initial Select Training will set the initial sequence of Mode Flag column entries to TRAIN, indicating training, up to the percentage threshold. The remaining entries will be set to TEST. For example, if there are 50 rows of training data and the setting is 80%, then the first 40 rows will be defined as training data, with the remainder defined as testing data. Initial Select Testing performs a similar function, but the initial rows are set to TEST, corresponding to testing data, and the remaining rows are set to TRAIN.

Random Select Data will randomly assign rows to be TRAIN or TEST, with the percent setting governing the approximate percentage of data to be set as training data. For example, a setting of 90% will generate a random classification with approximately 90% of the data classified as training data.

Auto Select Data will use a heuristic method to assign rows to be TRAIN or TEST, with the percent setting governing the approximate percentage of data to be set as training data. The method will scan for those rows which contain extremes or large variations of values in each column and include additional rows with data that may be suitable as possible, depending on the percent setting.

Invert Current Mode will scan any pre-existing classifications and invert them. For example, a sequence of 50 rows that had the first 40 set to TRAIN and the last ten set to TEST, would be inverted so that the first 40 were set to TEST and the last ten were set to TRAIN. If the Mode Flag column entry for a row is not filled in, then the setting will become TEST since a non-existent entry corresponds to an implied TRAIN.

Warnings: **Select Data Mode** offers some of the more useful operations on the classification of data between training and testing. However, its options may not always be suitable. When none of the **Select Data Mode** options are suitable or desirable, it is always possible to enter the classification directly in the Mode Flag column.

Auto Select Data is currently implemented as an Excel macro sequence and could take large amounts of time to classify larger data sets.

8.1.10 Edit Mode Lists



Figure 8-6 Edit Mode Lists Dialog Box

Function: **Edit Mode Lists** is used to enter or change the Symbol List, Min limit, or Max limit, for an Input or Target column. The Symbol List, Min limit, or Max limit, entered for a Target column is applied to an Output column.

Usage: To use **Edit Mode Lists**, select the column that is to have a Symbol List, Min limit, or Max limit, created or updated and execute the command. A dialog box will appear with fields for a Symbol List, Min limit, or Max limit, present or grayed out depending on the settings of the MIN, MAX, or SYMBOL, mode rows. Each field, if present, is clear if the the field has not been defined previously, or is initialized to the old settings if they have been defined previously. Enter or edit the settings in numeric or symbolic form, while maintaining consistency between the fields. The Symbol List, Min limit, or Max limit, can be accepted with **OK** or aborted with **Cancel**.

Effect: If symbolic data is entered or edited in the Symbol List field and the User Set option is enabled, then the Symbol List and the symbols defined for Min and Max, if Min and Max fields are present, are entered in the respective SYMBOL, MIN, and MAX, fields of the column selected. If symbolic data is entered or edited in the Symbol List field and the Auto Set option is enabled, then the Symbol List, and the first symbol and last symbol of the Symbol List, if Min and Max fields are present, are entered in the respective SYMBOL, MIN, and MAX, fields of the column selected. If the Symbol List is left blank and the User Set option is enabled, then the Min and Max values, if Min and Max fields are present, are entered in the respective MIN and MAX fields of the column selected. If the Symbol List is left blank and the Auto Set option is enabled, then the Min and Max values scanned from the designated row range of the designate column, if Min and Max fields are present, are entered in the respective MIN and MAX fields of the column selected. Any entries in the Min and Max fields, whether numeric or symbolic, are ignored in Auto Set mode.

The Symbols in the Symbol List define the valid symbolic values to be used as Inputs, Targets or Outputs, depending on the column type. The Symbols in the Symbol List will be assigned values that are increasing in proportion to their position in the list. Thus for the list <RED, GREEN, BLUE>, RED will be assigned the lowest value, GREEN will be assigned an intermediate value, and BLUE will be assigned the highest value. The same Symbol in different Symbol Lists are treated independently and are not assigned the same value. Thus the two lists, <RED, GREEN, BLUE> and <GREEN, BLUE, VIOLET>, have common Symbols, GREEN and BLUE, but they are not equal between Symbol Lists.

Warnings: It will not matter which rows in the column are used to select the column; only the fields defined by the intersection of the selected column and the defined Mode Flag rows will be set. The Mode Flag column and the Mode Flag rows must have been set previously with the **Set Mode Flag Column** and **Set Mode Rows** commands. The entries must be consistent, that is all symbolic or all numeric, not mixed symbolic and numeric.

The symbols of a Symbol List can be any alphanumeric combination separated by commas. Leading white space, space or tab, in a symbol is ignored, but trailing white space is distinctive. The symbols defined

in the Min and Max fields, if present, must match one of the defined Symbols and the Min symbol must precede the Max symbol in the Symbol List. Most neural networks are not able to accurately resolve more than 5-10 Symbols for an Input or Target. There is a limit of 255 characters in a Symbol List.

8.1.11 Set Network Size

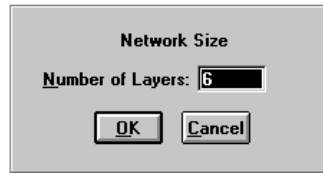


Figure 8-7 Network Size Dialog Box

Function: **Set Network Size** is used to define the structure of the neural network used in the problem, in particular the number of hidden layers and the number of neurons per layer.

Usage: To use **Set Network Size**, make sure that all rows and columns are first defined, then execute the command. A dialog box will appear asking for the Number of Layers in the neural network. There is a default minimum of 2 layers, the input and output layers. To add one or more hidden layers, you would change the entry and confirm **OK**. Another dialog box will then appear allowing you to define the number of neurons in each hidden layer which can be confirmed with **OK**. Changes can be aborted at either step with **Cancel**.

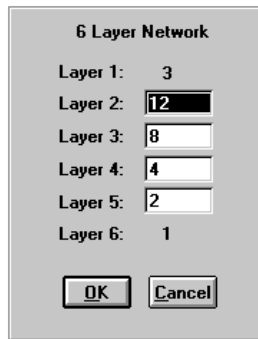


Figure 8-8 Network Description Dialog Box

Effect: When the first dialog box, Neural Network Configuration, is presented, you can specify the total number of layers in the neural network. Since the input and output layers are required layers, the number entered is 2 plus the number of desired hidden layers; for example, 4 would specify the default layers plus 2 hidden layers. The Network Description Dialog Box will then allow the specification of neurons for each layer. The input and output layer neurons are fixed by the number of Input columns and Output columns specified previously. Each hidden layer must have at least one neuron, but the upper limit is defined by a combination of memory availability and processing time you are willing to devote to the problem. Practical experience suggests using a range from a number midway between the preceding and succeeding layers to double the number of neurons in the preceding layer. For example, in a 3 layer network with 12 inputs and 2 outputs, a workable range is from 6 to 24 neurons for the one hidden layer. Once the network configuration has been confirmed, Neuralyst will build the neural network and initialize the starting weights for the neural network.

Warnings: **Set Network Size** is usually the last configuration step performed; in particular all columns must be defined before it can be executed. **Set Network Size** must also be executed any time there is a change in the number of columns in any column lists. The network configuration may be changed through another **Set Network Size** at any time, but when this is done, the previous weights, representing any learning that has taken place to that point, will be overwritten

and so forgotten. In that situation, Neuralyst will warn you that learning will be forgotten, allowing you to confirm your intentions.

8.2 Neural Network Operations Menu

| Neural | |
|----------------------------|----|
| Reload Network | ^l |
| Train Network | ^t |
| Run/Predict with Network | ^r |
| Run Genetic Supervisor... | ^g |
| Set Network Parameters... | ^p |
| Set Enhanced Parameters... | |
| Set Genetic Parameters... | |
| Plot Training Error | |
| Reset Weights... | |
| Histogram Weights | |
| Unpack Weights | |

Figure 8-9 Neural Menu

The **Neural** menu provides you with access to the commands necessary to control the operation of Neuralyst once a problem structure and corresponding neural network have been defined.

8.2.1 Reload Network

Function: **Reload Network** allows saved Neuralyst worksheets to be reloaded. It also allows switching between two or more Neuralyst worksheets that may be open at the same time.

Usage: **Open** a Neuralyst worksheet or bring an opened Neuralyst worksheet to the front and make it the active worksheet, then execute **Reload Network**. The worksheet will now be ready to run Neuralyst.

Effect: **Reload Network** will cause the configuration information and network weight values saved in the Working Area of a Neuralyst worksheet to be loaded. Once a Neuralyst worksheet is loaded the configuration may be changed or the neural network can be trained, tested or run.

Warnings: Neuralyst remembers which worksheet was last loaded and will require that worksheet be the active worksheet when it runs; this helps prevent mistakes and confusion when multiple Neuralyst worksheets are active. If you are not sure which worksheet is active or if you are not sure that a change you have made to the network configuration has taken effect, it doesn't hurt to do a **Reload Network**. If you change any data in the Problem Definition Area, even if the network configuration has not changed, a **Reload Network** should be done since Neuralyst keeps a copy of all data and needs to be informed so it can refresh its copy.

8.2.2 Train Network

Function: **Train Network** causes Neuralyst to use the set of defined training patterns as inputs to the neural network, backpropagating the resulting errors (differences between actual outputs and targets) to adjust the network weights.

Usage: To use **Train Network**, once the pattern and network configurations have been defined with the **Config** menu or a saved network has been reloaded, execute the command. Once training is started, it may be halted and Neuralyst returned to command mode by using the **Esc** {**Esc** or **cmd-.**} key.

Effect: **Train Network** will cause Neuralyst to sequence through the set of training patterns by shifting the defined pattern window through the defined data. As the pattern window is shifted, Neuralyst will submit the pattern to the neural network for training if the Mode Flag is TRAIN for that pattern window. The sequence will recycle to the beginning of the defined data once the end of the defined data is reached. Each complete pass through the data is called a training epoch. After the number of epochs specified in the Epochs per Update of the **Set Network Parameters** command, the Statistics Block is updated.

This process will continue until you cause it to halt by typing the **Esc** {**Esc** or **cmd-.**} key, until the neural network achieves 100% output accuracy with respect to the defined Training Tolerance, or reaches the Epoch Limit specified with the **Set Network Parameters** command.

Warnings: Once training begins, the network weights will be modified from whatever values were saved previously. The Output columns will only be valid for those rows that correspond to TRAIN entries in the Mode Flag column or for all rows if no Mode Flag Column has been set.

8.2.3 Run/Predict with Network

Function: **Run/Predict with Network** causes Neuralyst to use the set of defined testing or running patterns as inputs to the neural network. No backpropagation takes place so the network weights are not modified.

Usage: To use **Run/Predict with Network**, once the pattern and network configurations have been defined with the **Config** menu or a saved network has been reloaded, execute the command. Testing or running will proceed through the end of the defined data.

Effect: **Run/Predict with Network** will cause Neuralyst to sequence through the set of testing/running patterns by stepping the defined pattern window through the defined data. As each pattern window is stepped, Neuralyst will submit the pattern to the neural network for testing or running if the Mode Flag is TEST for that pattern window. The sequence will stop once the end of the defined data is reached. For testing data, the output accuracy with respect to the defined Testing Tolerance is reported in the Statistics Block. For running data (no comparison to targets), the Statistics Block will change but the statistics are not meaningful.

Warnings: Testing or running does not modify the network weights. The Output columns will only be valid for those rows that correspond to TEST entries in the Mode Flag column or for all rows if there is no Mode Flag Column set.

8.2.4 Run Genetic Supervisor

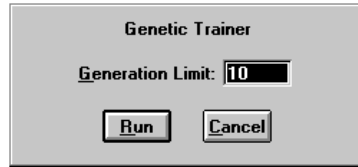


Figure 8-10 Genetic Trainer Dialog Box

Function: **Run Genetic Supervisor** causes Neuralyst to use the parameters subject to variation as defined in **Set Genetic Parameters** combined with the definition of the neural network model and iterate through a selection of candidate solutions to identify an optimal neural network.

Usage: To use **Run Genetic Supervisor**, first perform all the steps necessary to define a neural network to the stage ready for training; then select the desired parameters to control the genetic optimization process in **Set Genetic Parameters**; finally select **Run Genetic Supervisor** in the **Neural** menu and execute the command. A dialog box will appear allowing the specification of the number of generations to iterate for the genetic optimization process. Enter the number of desired generations and start the process with **OK** or abort with **Cancel**. At the conclusion of the desired number of generations, the best candidate solution will be reported and it may be retrieved to overwrite the existing neural network model by accepting it with **Retrieve** or rejecting it with **Cancel**.

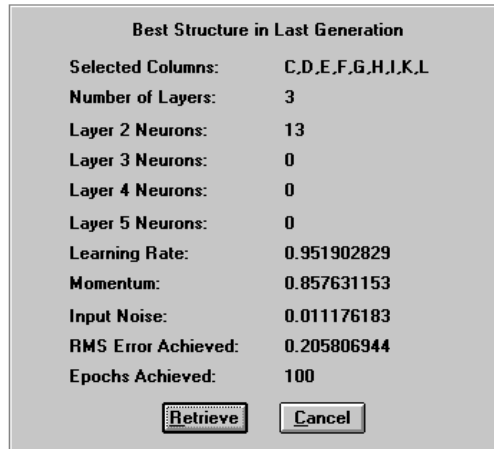


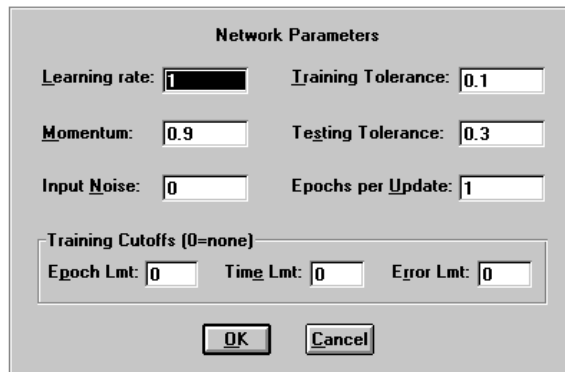
Figure 8-11 Best Structure

Effect: **Run Genetic Supervisor** will cause Neuralyst to make a copy of the Input and Target data. It will then create a population of structures according to the parameters set in **Set Genetic Parameters** and evaluate them against the Fitness Criteria and Fitness Limit set. The population will be evolved over the number of generations selected. The current generation, current structure, and best fitness values for the current generation are reported regularly. At the designated generation, the best candidate solution will be reported and it may be used to replace the current neural network model.

Warnings: **Run Genetic Supervisor** makes a copy of all Input and Target data as well as keeping its own data structures. As a result, using the Genetic Supervisor will approximately double the amount of memory that is normally used by Neuralyst. The Genetic Supervisor has to keep track of both the original neural network model and numerous variants; as a consequence there are many operations that can be performed which will invalidate its copy of the various states it is attempting to track, resulting in a need to reinitialize its state. Its state will always be valid as long as only **Set Genetic Parameter** settings for Population Mode, Genetic Operators, or Fitness Criteria are adjusted; most other commands in the **Config** menu, **Set Network Parameters**, or **Set Enhanced Parameters** will result in a reinitialization being

needed. The genetic optimization process will stop because it is interrupted or because it reaches the designated generation. A repetition of the **Run Genetic Supervisor** command will allow the genetic optimization process to continue from its last point if it was interrupted or to a higher generation if the generation setting is increased; so long as only the previously indicated acceptable operations are performed between **Run Genetic Supervisor** commands. The Genetic Supervisor will overwrite the original neural network model if the best candidate solution is retrieved; if the original neural network model is still desired, a copy of the worksheet should be made and saved before **Run Genetic Supervisor** is executed.

8.2.5 Set Network Parameters



The screenshot shows a dialog box titled "Network Parameters". It contains the following fields and values:

- Learning rate: 1
- Training Tolerance: 0.1
- Momentum: 0.9
- Testing Tolerance: 0.3
- Input Noise: 0
- Epochs per Update: 1

Below these fields is a section titled "Training Cutoffs (0=none)" which contains three sub-fields:

- Epoch Lmt: 0
- Time Lmt: 0
- Error Lmt: 0

At the bottom of the dialog are two buttons: "OK" and "Cancel".

Figure 8-12 Network Parameters Dialog Box

Function: **Set Network Parameters** allows various parameters affecting the network training, testing, and computation process to be adjusted.

Usage: To use **Set Network Parameters**, execute the command. A dialog box, Network Parameters, showing the current values of nine control parameters, Learning Rate, Momentum, Input Noise, Training Tolerance, Testing Tolerance, Epochs per Update, Epoch Limit, Time Limit, and Error Limit, will be displayed. These values may be changed and confirmed with **OK** or aborted with **Cancel**.

Effect: **Set Network Parameters** will allow you to change one or more of nine parameters, Learning Rate, Momentum, Input Noise, Training Tolerance, Testing Tolerance, Epochs per Update, Epoch Limit, Time Limit, and Error Limit, that affect the training process.

Learning Rate determines the magnitude of the correction term applied to adjust each neuron's weights when training. Learning Rate corresponds to the variable **LR** in Equation 3-3. Learning Rate must be positive, is adjusted in the range of 0 to 1 and has a default value of 1. Large values of Learning Rate will cause the network to train more quickly, but too large a value may cause the training to be unstable and no learning will occur.

Momentum determines the “lifetime” of a correction term as the training process takes place. Momentum corresponds to the variable **M** in Equation 3-6. Momentum must be greater than or equal to 0 but less than 1 and has a default of 0.9. Values of Momentum closer to 1 will cause the neural network to retain more of the impact of previous corrections to the current corrections. Values of Momentum close to 0 will allow mostly or only the current corrective term to have an effect. Momentum helps to smooth out the training process so that no single aberrant instance can force learning in an undesirable direction.

The Input Noise parameter enables the addition of noise during the training process. Input Noise has a value of 0 as a default, meaning no noise is added. Input Noise should be set between 0 and 1, meaning 0% to 100% of the input range will be set as a noise range. For example, if the input ranges from 0 to 15 and the Input Noise is set to 0.1, then random values ranging from 0 to 1.5, 10% of 15, will be added to or subtracted from each input value. Input Noise is only in effect when training.

The Training Tolerance and Testing Tolerance parameters define the percentage error allowed in comparing the neural network output to the target value to be scored as “Right” during training and testing, respectively. The Tolerance parameters should be between 0 and 1, with Training Tolerance having 0.1, or 10%, as a default and Testing Tolerance having 0.3, or 30%, as a default. The Tolerance parameters are defined as a percentage of the range between the highest and lowest values in the Target columns. For example, if values in the

Target columns ranged from 100 to 300, a Tolerance of 0.2, corresponding to 20%, would allow errors of 20% of 200 (300 minus 100) or 40. The Training Tolerance value has no effect on the learning algorithm. However, when Neuralyst finds 100% Right, as defined by Training Tolerance, it will automatically stop training. The Testing Tolerance value is used only for scoring during testing or running.

The Epochs per Update parameter allows you to control the number of epochs between updates of the neural network results which are displayed in the Network Run Statistics block in the worksheet. Epochs per Update has a default value of 1. However larger values will mean less frequent communication between the neural network and the worksheet, reducing overall training time.

The Epoch Limit parameter sets a maximum number of training epochs the neural network will undergo in those situations where you wish to control the number of training epochs rather than setting a Training Tolerance. Epoch Limit has a default value of 0, which means that there is no limit set.

The Time Limit parameter sets a maximum number of hours (enter minutes as a decimal fraction of an hour, for example 0.25 would be $\frac{1}{4}$ hour or 15 minutes) to continue training. Time Limit has a default value of 0 which means that there is no limit set.

The Error Limit parameter sets a maximum increase allowed, in the RMS Error values of the training data or the testing data, from the lowest achieved value of either. Under normal circumstances, a neural network that is training properly will steadily decrease the RMS Errors of each set of data. However, if the neural network exceeds its capacity or starts experiencing some effects of overtraining, some epochs may increase the RMS Error of either or both sets of data. Setting an Error Limit value will stop the training process if the RMS Error of either set increases, over any number of epochs, more than the amount set. Error Limit has a default value of 0 which means that there is no limit set.

When more than one of Epoch Limit, Time Limit, or Error Limit, is set, whichever limit is reached first will terminate the training process. When no limit is set or if limits are set but the limit condition is never reached, the normal termination condition occurs when the

neural network outputs meet the Training Tolerance criteria for all training data.

Warnings: It is important to observe the limitations on the range of parameter values for each parameter, though Neuralyst will check for valid values. It is possible to set Learning Rate, Momentum, and Input Noise parameters to values that cause little or no learning to take place; you should observe the change in RMS Error to be sure that it is being reduced at a satisfactory rate.

The time measured in the Time Limit parameter is actual neural network computation time — not wall-clock or waiting time. Thus parameter settings, for example low values of Epochs per Update, which result in heavy I/O, without computation, will appear to run for longer than the set Time Limit.

The RMS Error of the testing data is evaluated every training epoch if Error Limit is set to non-zero values. To skip this evaluation cost, particularly with large neural networks that take longer to train, set Error Limit to zero. With a zero setting, testing data will not be evaluated, saving substantial processing time. However, this also means that the RMS Error plot of the testing data will not be available in the **Plot Training Error** command.

8.2.6 Set Enhanced Parameters

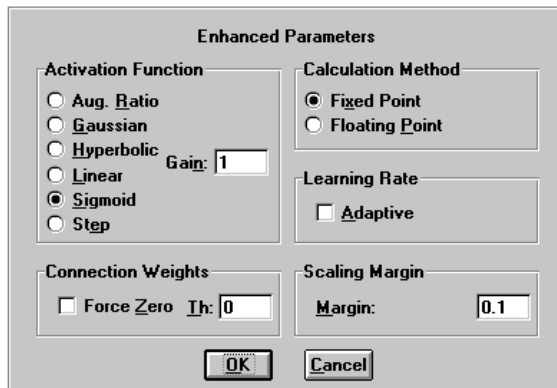


Figure 8-13 Enhanced Parameters Dialog Box

Function: **Set Enhanced Parameters** allows several parameters

controlling special functions and modes, that can modify the behavior of the neural network, to be selected.

Usage: To use **Set Enhanced Parameters**, execute the command. A dialog box, Enhanced Parameters, showing the current values of five parameters, Function, Function Gain, Force Zero, Zero Threshold, and Adaptive Learning Rate, will be displayed. These values may be changed and confirmed with **OK** or aborted with **Cancel**.

Effect: **Set Enhanced Parameters** will allow you to change one or more of five parameters, Function, Function Gain, Force Zero, Zero Threshold, and Adaptive Learning Rate, that change the behavior of the neural network.

The Function parameter allows a change of the neuron activation function described in Section 3.3 and Section 6.6. The functions that are available are: Gaussian, Linear, Sigmoid, Step, Hyperbolic and Augmented Ratio. The Gaussian function is a smooth, differentiable function which transforms values within a range close to 0 to +1, very large negative or positive values to 0, and transforms intermediate values with varying proportion. The Linear function provides a continuous response multiplying input values by a constant scaling factor until the limits of 0 or +1 are reached. The Sigmoid function is also a smooth, differentiable function, which forces large negative or large positive values to 0 or +1, respectively, but transforms intermediate values with varying proportion. The Step function is a sharp switching function, all negative inputs are forced to 0 and all positive inputs are forced to +1. The Hyperbolic function is another smooth, differentiable function, which forces large negative or large positive values to -1 or +1, respectively, but transforms intermediate values with varying proportion. The Hyperbolic function is actually the Hyperbolic Tangent function and is very similar to the Sigmoid function, except it is negative-going. The Augmented Ratio function is a smooth, differentiable function which transforms values within a range close to 0 to 0, very large negative or positive values to 1, and transforms intermediate values with varying proportion. The Augmented Ratio function is similar to the Gaussian function, but inverted. Thus selecting different neuron functions can significantly change the behavior of the neuron.

The Function Gain parameter allows a change in the scaling or width of the selected Function. Increasing Function Gain will narrow the central region in the Gaussian function or steepen the slope of the Linear function and the Sigmoid function. Decreasing the Function Gain, correspondingly, will broaden the central region in the Gaussian Function or make the slope of the Linear or Sigmoid functions shallower. With a Function Gain setting of 1, all three functions will cover the normal input range of the neural network smoothly. The Function Gain applies to all of the Activation Functions except the Step function.

The Force Zero parameter enables a special mode which scans the weights after every training epoch and sets those weights which are close to 0 to be 0. The Zero Threshold parameter selects the threshold below which weight values are set to 0. Thus, a Zero Threshold value of 0.01 with Force Zero enabled will cause a weight value of 0.009 to be made to be 0, while a value of 0.15 will be left unchanged.

The Calculation Method parameter allows a selection of Fixed Point calculation or Floating Point calculation. Fixed Point mode is a very fast and memory efficient technique, allowing speeds up to 2-3x a computer equipped with a math co-processor using Floating Point calculations. It is inherently less precise, but is satisfactory for the majority of neural network problems that require precision to only 2-3 places. Floating Point mode is more precise and has higher resolution, but requires a math co-processor — not present on many 386, 486SX, 486SL, 68030, and 68LC040 processors — more memory, and more processing time. Fixed Point mode was the native mode of earlier versions of Neuralyst; but now either mode can be selected depending on the circumstances.

The Adaptive Learning Rate parameter enables a special mode which evaluates the RMS Error after every training epoch and causes a revision of the Learning Rate for the next training epoch. If the RMS Error is high, then a high Learning Rate will be set, as the RMS Error is reduced, the Learning Rate will be reduced correspondingly. The net effect is to speed up the training process when the neural network is far away from the correct weights, but to slow it down as it gets closer.

The Scaling Margin parameter adds additional headroom, as a percentage of range, to the rescaling computations used by Neuralyst in preparing data for the neural network or interpreting data from the neural network. All Input, Target, and Output data actually processed by the neural network must be in the range 0 to 1. As a consequence, all data presented or interpreted is rescaled from the actual worksheet values to values in the range of 0 to 1. Setting Scaling Margin above 0, increases the amount of headroom or margin, decreasing the range onto which an Input, Target, or Output, is mapped. As an example, if the Scaling Margin is set to 0.1, or 10%, then 10% margin is allowed and split between the high end and the low end of the range, resulting in a mapping to the range 0.05 to 0.95 rather than 0 to 1. This margin can facilitate precision in problems that have continuously variable data or that may exceed the current values in the future by a small amount. Scaling Margin modifies the Min or Max limits that may be set for an input or target by the designated amount.

The options of Input Noise, Force Zero, and Adaptive Learning Rate will typically increase the total number of training epochs it takes to achieve a successfully trained neural network over a “no-frills” approach. However, when used judiciously, the quality of training - and the corresponding quality of predictions — of the neural network can be notably improved. Fixed Point calculation mode generally provides the fastest training; however, for continuously variable inputs or outputs requiring high resolution, Floating Point calculation mode can be more precise. In addition, on high performance Pentiums or PowerPCs, Floating Point approaches Fixed Point speed. Scaling Margin is another control which is useful for problems that have continuously variable inputs or outputs to allow higher precision.

Warnings: Selecting neuron activation functions aside from the Sigmoid will typically change the behavior of the neural network significantly. The Hyperbolic function is probably the most useful after the Sigmoid. However, the Hyperbolic function is only effective in Floating Point calculation mode. The Step function has limited utility and is particularly difficult to train and has been included mostly for experimental purposes for readers interested in duplicating the work of early researchers on neural networks. The Linear,

Gaussian, and Augmented Ratio functions may or may not train well, depending on the characteristics of the training data.

While there are no limits on the settings of the Function Gain parameter or the Zero Threshold parameter, extreme values will prevent training. Typically values of Function Gain should not exceed the range 1/10 to 10 and the Zero Threshold parameter should not exceed a level of 0.5.

Selecting Force Zero and Adaptive Learning Rate (as well as Input Noise in the **Set Network Parameters** command) will typically increase the total number of training epochs it takes to achieve a successfully trained neural network over a “no-frills” approach.

Setting Scaling Margin to allow headroom for future values that exceed the current data range is a marginal practice. Since, in principle, the neural network has not been trained on data outside the current range, it cannot be expected to accurately predict data outside the current range, even with headroom. In practice, it can be effective for certain neural network models.

8.2.7 Set Genetic Parameters

| Genetic Training Parameters | | |
|-----------------------------|--|--|
| Input Data | Training Parameters | Genetic Operators |
| Inclusion Rate: 0.75 | Learning Rate: 0.5 | Crossovers: 1 |
| | Momentum: 1 | Mutation Rate: 0.1 |
| | Input Noise: 0.03 | |
| Network Structure | Population Control | Fitness Criteria |
| Max Layers: 4 | Pool Size: 3 | <input type="radio"/> Train Epochs |
| L2 Neuron Lmt: 30 | <input type="radio"/> Closed Pool | <input checked="" type="radio"/> Train Error |
| L3 Neuron Lmt: 10 | <input checked="" type="radio"/> Immigration | <input type="radio"/> Test Epochs |
| L4 Neuron Lmt: 0 | <input type="radio"/> Emmigration | <input type="radio"/> Test Error |
| L5 Neuron Lmt: 0 | | Fitness Limit: 100 |
| | OK | Cancel |

Figure 8-14 Genetic Parameters Dialog Box

Function: **Set Genetic Parameters** allows several parameters

controlling the creation of structure populations, the manner in which they are evolved, and the way they are evaluated, to be selected.

Usage: To use **Set Genetic Parameters**, execute the command. A dialog box, Genetic Parameters, showing the current values of the 15 parameters, Inclusion Rate, Max Layers, four Layer **n** Neuron Limits, Learning Rate, Momentum, Input Noise, Pool Size, Pool Mode, Crossovers, Mutation Rate, Fitness Criteria, and Fitness Limit, will be displayed. These values may be changed and confirmed with **OK** or aborted with **Cancel**.

Effect: **Set Genetic Parameters** will allow you to change one or more of the fifteen parameters, Inclusion Rate, Max Layers, four Layer **n** Neuron Limits, Learning Rate, Momentum, Input Noise, Pool Size, Pool Mode, Crossovers, Mutation Rate, Fitness Criteria, and Fitness Limit, that bound the limits of the genetic optimization process and change the behavior of the Genetic Supervisor.

The Input column Inclusion Rate controls the average rate of Input column inclusion in structure initialization. The user may select a nominal percentage from 1 to 100% which represents the average rate of inclusion. If the user wants to force all Input columns to be included, then the user can set 100%. The default setting is 75%.

The first and last layer of each neural network is automatically determined by the number of inputs and outputs defined for the original neural network model. Within those constraints, the optimized neural network can have from two to six layers and a wide range of variations in number of neurons per layer. The user can constrain the maximum number of layers and the maximum number of neurons per hidden layer. An implicit rule is that if a lower hidden layer is zero, then higher hidden layers must be zero. The default maximum settings are 4 layers, an input layer, 2 hidden layers, and an output layer; with 30 neurons in the first hidden layer and 10 neurons in the second hidden layer.

There are three network parameters which can be optimized by the Genetic Supervisor: Learning Rate, Momentum, and Input Noise. Learning Rate and Momentum can vary from almost 0 to 1, while the Input Noise can vary from 0 to 0.1. The user can constrain the minimum value of Learning Rate, the maximum value of Momentum, and the maximum value of Input Noise. The default values are a

minimum of 0.5 for Learning Rate, a maximum of 1 for Momentum, and a maximum of 0.03 for Input Noise.

There are two controls which determine the management of populations. The first is a control which sets the total population size. Changing this sets the number of structures initialized or evaluated in each generation. The second is a control which chooses among the three modes Closed Pool, Immigration, or Emigration. With Closed Pool set, then the initial population pool will only be cross-bred or mutated with no new structures initialized. With Immigration set, the population pool will be evolved and every generation new structures will replace the weakest structures of the existing pool. With Emigration set, the population pool will be evolved and every generation the best structure will be emigrated to an entirely new population. The default settings are for population size to be 3 and Immigration to be enabled.

There are two settings the user can control to determine the mechanism for genetic optimization. These settings are Crossovers and Mutation Rate. Cross-breeding is controlled by the Crossovers setting, which determines the amount of intermingling of features on the same string to create new structures. A setting of 1 means that two strings are crossed over at one point; a setting of 2 means that two strings are crossed over at two points. The maximum setting for Crossovers is 10 and the default is 1. The secondary mechanism for creating new structures is mutation. With mutation, structures are chosen at random, then features are randomly changed to new values. The mutation rate sets the percentage of structures which will undergo a mutation rather than a cross-breeding to create the new population. The maximum setting for mutation is 100%. Mutation is always in place of cross-breeding; the two never occur at the same time on a single structure. The default setting for mutation rate is 10%.

The evaluation of each structure is based training or testing while comparing the best RMS Error level achieved or the least number of epochs. There are four modes which can be set: Train Epochs, Train Error, Test Epochs, and Test Error. Train Error finds the structure with the least RMS Error while holding epochs to the set Fitness Limit. Train Epochs finds the structure with the least epochs to achieve the RMS Error level set in Fitness Limit. These two operate on training data only. Test Error finds the structure with the least

RMS Error in the test set data while holding epochs to the Fitness Limit. Test Epochs finds the structure with the least epochs to achieve the RMS Error level of the test set data as set in Fitness Limit. These two measure RMS Error by evaluating the testing data. The default Fitness Criteria mode is Train Error with Fitness Limit set to 100 epochs. This will optimize RMS Error while training each candidate solution to an epoch limit of 100.

Warnings: Setting a low value of Inclusion Rate and a small Pool Size may result in very few Input columns being evaluated. The Neuron Limits for hidden layers in excess of those specified by the Max Layers settings must be 0. A very high Mutation Rate means that relatively little cross-breeding occurs, since mutation supersedes cross-breeding. Train Epoch and Test Epoch optimize for least epochs, so the Fitness Limit represents the reference RMS Error settings and must be between 0 and 1 for these Fitness Criteria. Train Error and Test Error optimize for least RMS Error, so the Fitness Limit represents the reference epoch count and must be an integer greater than 1 for these Fitness Criteria.

Many **Config** menu, **Set Network Parameters**, and **Set Enhanced Parameters**, commands will invalidate the Genetic Supervisor state and require another **Set Genetic Parameters** command to reinitialize the Genetic Supervisor. This will cause genetic optimization results to be lost.

8.2.8 Plot Training Error

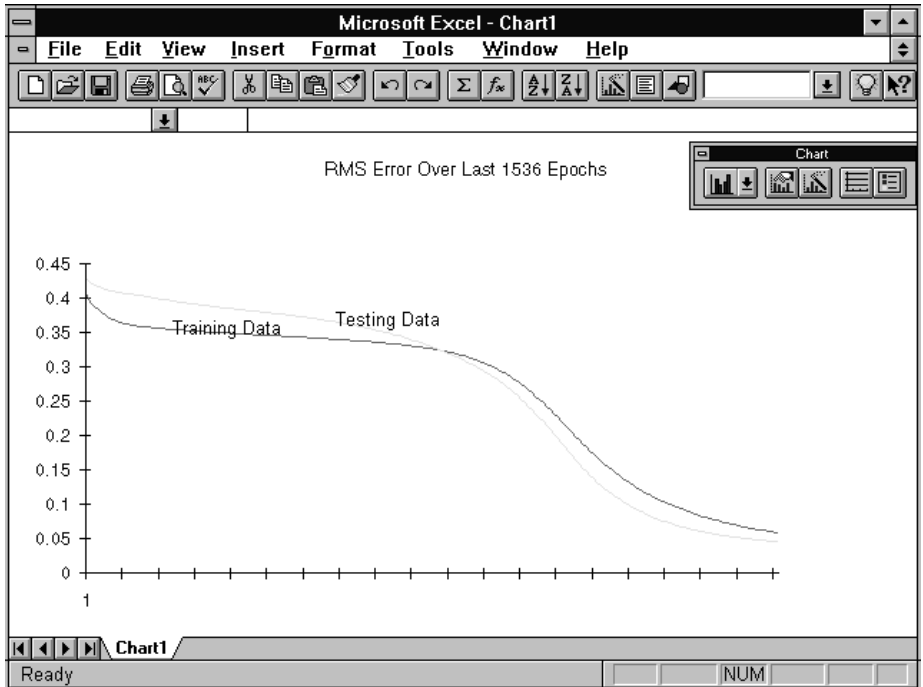


Figure 8-15 Training Error Chart

Function: **Plot Training Error** will cause a line plot of the RMS Error of the training data and the testing data over the recent training epochs to be displayed in an Excel chart.

Usage: To use **Plot Training Error** execute the command. Neuralyst will then transfer the set of values of RMS Errors of the training data and the testing data resulting from training to that point and create a new chart window containing line plots of the data.

Effect: **Plot Training Error** will display plots of the convergence of the RMS Error values over the past training epochs of the neural network. One plot will indicate the convergence of the neural network on achieving learning of the training data. The second plot will show the success of the neural network on the testing data while training is occurring. The second plot will only be available if there is testing data and if the Error Limit (in **Set Network Parameters**) has been

set to be non-zero. This command is executed when you want to get an impression of how much progress has been made and how successful the neural network has been at learning during training.

Ideal plots of the RMS Error of the training data will show a steep descent early on with progress in further reductions of RMS Error, though not as steep as initially, as training progresses. If the RMS Error attains a long term plateau, then it typically indicates that the neural network has achieved as much learning as it can. If the RMS Error starts increasing, it often indicates that the neural network is becoming severely overtrained. Along with the training data, the testing data will also be plotted (if settings permit). The RMS Error plot of the testing data can be used to measure the point at which the neural network training was most successful on testing data. Overtraining can often be seen earlier on this plot than for training data.

Warnings: The line plots generated by **Plot Training Error** are a snapshot of the RMS Error behavior up to the time the command was executed. Subsequent training will not be reflected in the chart. To view the effects of additional training, another **Plot Training Error** command must be given. In this event, new line plots are created, without effect on the previous line plots, allowing comparison of the effects of additional training epochs.

At approximately 250 epochs, and every doubling of epochs thereafter, the line plots will be rescaled — so plots at 100 epochs will have points for every epoch, but plots at 300 epochs will have only 150 points per plot (a point for every two epochs). Thus the scale will be different, though the shape and information will be consistent.

The training error history is not retained between reloads of neural network data. Thus if you have a partially trained neural network, save your work, reload the next day, train some more and then do a **Plot Training Error** command, the plot will only show the training error behavior over the most recent session. The training error history of the previous session will be lost and no longer available.

You will need to manually delete the line plots when you no longer want them.

8.2.9 Reset Weights



Figure 8-16 Weight Initialization Dialog Box

Function: **Reset Weights** provides for control of the weight initialization process.

Usage: To use **Reset Weights** execute the command. Neuralyst will provide a dialog box allowing for Auto or User Randomization of the initial weight values. The range of the initial weight values may also be set. Changes to the initialization options and the initialization operation itself can be confirmed with an **OK** or else the changes and operation may be aborted with a **Cancel**.

Effect: **Reset Weights** allows the neural network weights to be changed to new random values. (Neural network weights are initialized to random values to allow learning to occur without any established patterns or biases.) Prior to resetting the weights the user may set certain initialization options. These options are Auto Set Randomization, User Set Randomization, and Initial Limit. Once the options are set and the operation is confirmed, the weights are set to new initial values.

Selection of Auto Set will provide for a completely random set of weights. There will be no relationship between the values generated from one instance of Auto Set Randomization to the next or other instances of Auto Set randomization.

The user may select User Set Randomization in lieu of Auto Set Randomization, causing a particular predetermined and repeatable random sequence to be used, based on the value supplied

to User Set Randomization. In other words, supplying 1 to the User Set Randomization option will always cause the same random numbers to be used for the weights in a given neural network. Changing the value supplied will result in a new, but equally repeatable, set of random values to be generated. The initial value supplied to the User Set Randomization is 1 and may be changed to any other integer value.

With the Randomization type selected, the user may also set the Initial Limit value. This has a default value of 1, which means that initial weight values will be randomly chosen from the range -1 to +1. Reducing the Initial Limit value to 0.2, would result in initial weight values in the range -0.2 to +0.2. Initial Limit may be set to a number between 0 to 16, though initial values much larger than 1 are not very useful.

This command is executed when you want a fresh start to learning or training of the neural network. Weights are always randomized when set to starting values since fixed or ordered starting values may result in all neurons learning the same characteristic at the same time. Randomization allows individual neurons to become “sensitized” to different characteristics.

Warnings: All previous weights will be overwritten and so all prior learning will be forgotten when **Reset Weights** is executed. Because of this, Neuralyst will first ask you to confirm your intentions.

8.2.10 Histogram Weights

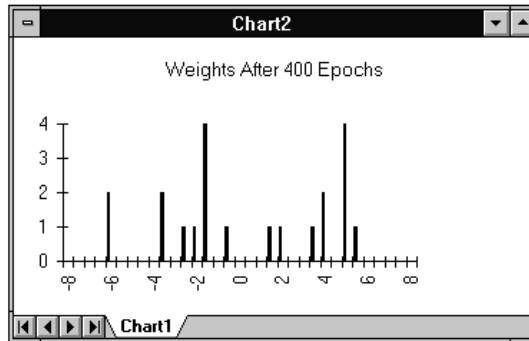


Figure 8-17 Weight Histogram Chart

Function: **Histogram Weights** will cause a histogram of the weights of the defined network to be displayed in an Excel chart.

Usage: To use **Histogram Weights** execute the command. Neuralyst will then read the weights of the neural network at that point and create a new chart window containing a histogram of the weight data.

Effect: **Histogram Weights** will display the distribution of weight values of the neural network. This command is executed when you want to get an impression of how much progress has been made and how successful the neural network has been at learning during training.

Early on, the weight values will be distributed in the range determined by the Initial Limit option in the Weight Initialization dialog box. If this is 1, then the histogram would show weight values distributed between -1 and +1. As time progresses, the distribution will spread out, with many values still in the initial range but with several values just outside the range and a few values at the extremes. This kind of distribution is most common in successfully trained neural networks.

If a neural network has been unsuccessful at training, a histogram of the weights will often show a distribution heavily biased to the

extremes, values below -8 and above +8. This may indicate a need to set a larger neural network or different network parameters.

Warnings: The histogram generated by **Histogram Weights** is a snapshot of the weights at the time the command was executed. Subsequent training will not be reflected in the chart. To view the effects of additional training, another **Histogram Weights** command must be given. In this event, a new histogram is created, without effect on the previous histogram, allowing comparison of the weight distribution from the two points of training. You will need to manually delete the histograms when you no longer want them.

8.2.11 Unpack Weights

The screenshot shows a Microsoft Excel window with the following data in the 'Unpacked Weights' worksheet:

| | A | B | C | D | E | F | G | H | I |
|----|------------------|----------|----------|----|----------|---|---|---|---|
| 1 | Unpacked Weights | | | | | | | | |
| 2 | | | | | | | | | |
| 3 | | Layer 2 | | | Layer 3 | | | | |
| 4 | N1 | -1.60693 | -1.63518 | N1 | -1.61473 | | | | |
| 5 | | 1.459551 | | | 4.870485 | | | | |
| 6 | | | | | 3.585479 | | | | |
| 7 | N2 | 5.143627 | 5.123119 | | -1.70692 | | | | |
| 8 | | -2.40044 | | | | | | | |
| 9 | | | | N2 | -3.54887 | | | | |
| 10 | N3 | 4.080883 | 4.071786 | | 1.839369 | | | | |
| 11 | | -5.96504 | | | 5.019307 | | | | |
| 12 | | | | | -3.25186 | | | | |
| 13 | | | | | | | | | |
| 14 | | | | N3 | -0.66611 | | | | |
| 15 | | | | | 5.503088 | | | | |
| 16 | | | | | -6.06227 | | | | |
| 17 | | | | | -2.08621 | | | | |
| 18 | | | | | | | | | |

Figure 8-18 Unpacked Weights Display

Function: **Unpack Weights** will cause the weights of the defined network to be displayed in a newly created Excel worksheet in a readable form.

Usage: To use **Unpack Weights** execute the command. Neuralyst will then write the unpacked weight values in a new Excel worksheet window.

Effect: **Unpack Weights** will cause a formatted display of the weight values to be written to a newly created Excel worksheet. This display is organized by neural network layers and by neurons in each layer, with each component being so labeled.

The first neural network layer displayed is Layer 2 (Layer 1 “neurons” are the inputs themselves by convention). The weights of each neuron in Layer 2 are shown in multiple columns grouped by neurons. The separate columns reflect the weights applied to each column designated as an input. The rows of the grouped columns reflect the different weights for each row used when multi-row inputs are defined. Thus the pattern window created by the number of inputs and the number of rows per pattern corresponds directly to the values shown in the columns and rows, respectively, for each neuron in the Layer 2 weight display. These weight values are outlined for each neuron. The threshold value for each neuron is outlined separately and terminates the weight values.

Subsequent neural network layers are displayed in a more simplified way with the weights of a neuron shown in single columns grouped by neurons. The first value in each column reflects the weight from the first neuron of the previous layer to that neuron, the second value in each column reflects the weight from the second neuron of the previous layer to that neuron, and so on. The column of weights is outlined for each neuron. The threshold value for each neuron is outlined separately and terminates the weight values.

Warnings: The display of weights generated by **Unpack Weights** is a snapshot of the weights at the time the command was executed. Subsequent training will not be reflected in the display. To view the effects of additional training, another **Unpack Weights** command must be given. In this event, a new worksheet will be created. You will need to manually delete the worksheets when you no longer want them.

8.3 Neuralyst Working Area

The Neuralyst Working Area is the area set aside for Neuralyst to save its internal data, save its control parameters and display its progress to you.

The Working Area is divided into several distinct areas: the Title Block, the Statistics Block, the Parameter Block, the Row Description Block, the Column Description Block, the Network Description Block, and the Network Weights Block. Also within the Working Area are three areas which are dedicated to the Genetic Supervisor: the Genetic Statistics Block, the Genetic Parameter Block, and the Genetic State Block.

Warning: Some fields in the Working Area depend on Excel calculations to be set to their proper values. If you have the Excel calculation mode set to Manual, instead of Automatic, then the information in these fields will not be properly updated. This may result in improper initialization or incorrect statistical updates while Neuralyst is executing.

It is generally best to have the Excel Calculation mode set to Automatic when working with Neuralyst. However, in some cases, particularly with large amounts of data or formulas, Excel is more responsive if Manual Calculation is set. In such cases, you may keep it set to Manual, switching to Automatic when you execute an operation from the Neuralyst **Config** menu or when actually training, testing, or running.

8.3.1 Title Block

The screenshot shows a Microsoft Excel window with the following content:

| | K | L | M | N | O | P | Q | R | S | T | |
|----|---|--|----------------------|---|---|---------------------|----------------|---|--------------------|---|--|
| 1 | | Neuralyst (TM) Version 1.4 | | | | | | | | | |
| 2 | | Copyright © 1994 Cheshire Engineering Corp. Portions Copyright © 1990-1994 EPIC Systems Co | | | | | | | | | |
| 3 | | | | | | | | | | | |
| 4 | | Network Run Statistics | | | | | | | | | |
| 5 | | 0.054459 | RMS Error | | | Row Information | | | Column Information | | |
| 6 | | 12 | Number of Data Items | | | 6 | First Row | | # Input Columns | | |
| 7 | | 12 | Number Right | | | 9 | Last Row | | 2 | | |
| 8 | | 0 | Number Wrong | | | 4 | Number of Rows | | Input Columns | | |
| 9 | | 100% | Percent Right | | | 1 | Rows/Pattern | | A | | |
| 10 | | 0% | Percent Wrong | | | 1 | Row Offset | | B | | |
| 11 | | 400 | Training Epochs | | | | | | A,B | | |
| 12 | | | | | | | | | | | |
| 13 | | Network Parameters | | | | Enhanced Parameters | | | | | |
| 14 | | 1 | Learning rate | | | Sigmoid Function | | | | | |
| 15 | | 0.9 | Momentum | | | 1 Function Gain | | | | | |
| 16 | | 0 | Input Noise | | | FALSE | Force Zero | | | | |
| 17 | | 0.1 | Training Tolerance | | | 0 | Zero Threshold | | | | |
| 18 | | 0.3 | Testing Tolerance | | | FALSE | Adaptive LR | | | | |

Figure 8-19 Title, Statistics & Parameter Blocks

The Title Block identifies the name of the program and provides the copyright notice for Neuralyst.

The Title Block is also used to carry the version number of Neuralyst that created and maintained the Working Area. This is useful when upgrading to new versions of Neuralyst. When a new version of Neuralyst is loaded with an old Neuralyst worksheet, it will check the version number and update the worksheet if possible or inform you of the need for conversion.

The Title Block also defines the top left corner of the Neuralyst Working Area. Any cell in the same row or higher and in the same column or higher is subject to modification by Neuralyst.

8.3.2 Statistics Block

The Statistics Block allows you to monitor the progress of training of the neural network and shows the success of prediction in test cases. The values in this block are updated for each cycle of the defined data during training or at the end of the defined data during testing or running.

RMS Error shows the *root mean square error* (the square root of the average of the error terms squared, a better measure than a simple average since errors of opposite signs cannot cancel each other) during training or testing between the outputs and the targets; it is not meaningful when running since there are no valid targets. When the neural network is training, the RMS Error should generally decrease at a steady rate, though there may be some short term fluctuations which cause small increases. If there are large fluctuations or the RMS Error is increasing, then some adjustment of Learning Rate or Momentum is needed through the **Set Network Parameters** command.

Number of Data Items represents the total count of cells in the Output columns or Target columns (remember Output columns are paired with Target columns) that are in the Problem Definition Area matching the current operating mode. For example, if there are two Output columns with 6 training rows (TRAIN flag) and 3 testing rows (TEST flag), then executing **Train Network** will cause this count to be 12 (2 columns times 6 rows) or executing **Run/Predict with Network** will cause this count to be 6 (2 columns times 3 rows).

Number Right represents how many of the data items counted in Number of Data Items have matching outputs and targets within the effective Tolerance parameter. Number Wrong represents those that are not within the effective, Training or Testing, Tolerance parameter. Percent Right and Percent Wrong are similar, except representing percentage values instead of counts.

When the network is training successfully, you will see these values change from predominantly wrong or a mix of right and wrong to mostly or all right. When the Number Right matches the Number of Data Items (the Percentage Right is 100%), then training will stop

automatically. When the network is testing, the Number Right and Number Wrong have similar meanings but in this case should be treated more as “scores” for the level of predictive success rather than a running indicator of learning progress. When the network is running, these values have no meaning.

The number of training epochs the neural network has undergone is recorded by the value of Training Epochs. In some circumstances, the user may prefer to limit training to a fixed number of epochs; Training Epochs provides this capability. In other circumstances, particularly when the user is trying to measure how well the neural network is learning under changing conditions, Training Epochs provides a reference point for evaluating training performance.

8.3.3 Parameter Block

The Parameter Block consists of two parts. The first part contains the current values of the Network Parameters. They may be changed through the **Set Network Parameters** command. The parameters displayed are Learning Rate, Momentum, Input Noise, Training Tolerance, Testing Tolerance, Epochs per Update, Epoch Limit, Time Limit, and Error Limit. The second part contains the current settings of the Enhanced Parameters. They may be changed through the **Set Enhanced Parameters** command. The parameters displayed are Function, Function Gain, Force Zero, Zero Threshold, and Adaptive Learning Rate.

Learning Rate determines the magnitude of the correction term applied to adjust each neuron’s weights when training. Learning Rate corresponds to the variable LR in Equation 3-3. Learning Rate must be positive, is usually adjusted in the range of 0 to 1 and has a default value of 1. Large values of Learning Rate will cause the network to train more quickly, but too large a value may cause the training to be unstable and no learning will occur.

Momentum determines the “lifetime” of a correction term as the training process takes place. Momentum corresponds to the variable M in Equation 3-6. Momentum must be greater than or equal to 0 but less than 1 and has a default of 0.9. Values of Momentum closer to 1 will cause the neural network to retain more of the impact of previous corrections to the current corrections. Values of

Momentum close to 0 will allow mostly or only the current corrective term to have an effect. Momentum helps to smooth out the training process so that no single aberrant instance can force learning in an undesirable direction.

The Input Noise parameter enables the addition of noise during the training process. Input Noise has a value of 0 as a default, meaning no noise is added. Input Noise should be set between 0 and 1, meaning 0% to 100% of the input range will be set as a noise range. For example, if the input ranges from 0 to 15 and the Input Noise is set to 0.1, then random values ranging from 0 to 1.5, 10% of 15, will be added to or subtracted from each input value. Input Noise is only in effect when training.

The Training Tolerance and Testing Tolerance parameters define the percentage error allowed in comparing the neural network output to the target value to be scored as “Right” during training and testing, respectively. The Tolerance parameters should be between 0 and 1; with Training Tolerance having 0.1, or 10%, as a default and Testing Tolerance having 0.3, or 30%, as a default. The Tolerance parameters are defined as a percentage of the range between the highest and lowest values in the Target columns. For example, if values in the Target columns ranged from 100 to 300, a Tolerance of 0.2, corresponding to 20%, would allow errors of 20% of 200 (300 minus 100) or 40. The Training Tolerance value has no effect on the learning algorithm. However, when Neuralyst finds 100% Right, as defined by Training Tolerance, it will automatically stop training. The Testing Tolerance value is used only for scoring during testing or running.

The Epochs per Update parameter allows you to control the number of epochs between updates of the neural network results which are displayed in the Network Run Statistics block in the worksheet. Epochs per Update has a default value of 1. However larger values will mean less frequent communication between the neural network and the worksheet, reducing overall training time.

The Epoch Limit parameter sets a maximum number of training epochs the neural network will undergo in those situations where you wish to control the number of training epochs rather than setting a

Training Tolerance. Epoch Limit has a default value of 0, which means that there is no limit set.

The Time Limit parameter sets a maximum number of hours (enter minutes as a decimal fraction of an hour, for example 0.25 would be $\frac{1}{4}$ hour or 15 minutes) to continue training. Time Limit has a default value of 0 which means that there is no limit set.

The Error Limit parameter sets a maximum increase in the RMS Error value from training epoch to training epoch. Under normal circumstances, a neural network that is training properly will steadily decrease the RMS Error. However, if the neural network exceeds its capacity or starts experiencing some effects of overtraining, some epochs may increase the RMS Error. Setting an Error Limit value will stop the training process if the RMS Error increases on any given epoch more than the amount set. Error Limit has a default value of 0 which means that there is no limit set.

When more than one of Epoch Limit, Time Limit, or Error Limit, is set, whichever limit is reached first will terminate the training process. When no limit is set or if limits are set but the limit condition is never reached, the normal termination condition occurs when the neural network outputs meet the Training Tolerance criteria for all training data.

The Function parameter allows a change of the neuron activation function described in Section 3.3. The functions that are available are: Gaussian, Linear, Sigmoid, Step, Hyperbolic and Augmented Ratio. The Gaussian function is a smooth, differentiable function which transforms values within a range close to 0 to +1, very large negative or positive values to 0, and transforms intermediate values with varying proportion. The Linear function provides a continuous response multiplying input values by a constant scaling factor until the limits of 0 or +1 are reached. The Sigmoid function is also a smooth, differentiable function, which forces large negative or large positive values to 0 or +1, respectively, but transforms intermediate values with varying proportion. The Step function is a sharp switching function, all negative inputs are forced to 0 and all positive inputs are forced to +1. The Hyperbolic function is another smooth, differentiable function, which forces large negative or large positive values to -1 or +1, respectively, but transforms intermediate values

with varying proportion. The Hyperbolic function is actually the Hyperbolic Tangent function and is very similar to the Sigmoid function, except it is negative-going. The Augmented Ratio function is a smooth, differentiable function which transforms values within a range close to 0 to 0, very large negative or positive values to 1, and transforms intermediate values with varying proportion. The Augmented Ratio function is similar to the Gaussian function, but inverted. Thus selecting different neuron functions can significantly change the behavior of the neuron.

The Function Gain parameter allows a change in the scaling or width of the selected Function. Increasing Function Gain will narrow the central region in the Gaussian function or steepen the slope of the Linear function and the Sigmoid function. Decreasing the Function Gain, correspondingly, will broaden the central region in the Gaussian Function or make the slope of the Linear or Sigmoid functions shallower. With a Function Gain setting of 1, all three functions will cover the normal input range of the neural network smoothly. The Function Gain applies to all of the Activation Functions except the Step function.

The Force Zero parameter enables a special mode which scans the weights after every training epoch and sets those weights which are close to 0 to be 0. The Zero Threshold parameter selects the threshold below which weight values are set to 0. Thus, a Zero Threshold value of 0.01 with Force Zero enabled will cause a weight value of 0.009 to be made to be 0, while a value of 0.15 will be left unchanged.

The Adaptive Learning Rate parameter enables a special mode which evaluates the RMS Error after every training epoch and causes a revision of the Learning Rate for the next training epoch. If the RMS Error is high, then a high Learning Rate will be set, as the RMS Error is reduced, the Learning Rate will be reduced correspondingly. The net effect is to speed up the training process when the neural network is far away from the correct weights, but to slow it down as it gets closer.

The Calculation Method parameter allows a selection of Fixed Point calculation or Floating Point calculation. Fixed Point mode is a very fast and memory efficient technique, allowing speeds up to 2-3x a computer equipped with a math co-processor using Floating Point calculations. It is inherently less precise, but is satisfactory for the

majority of neural network problems that require precision to only 2-3 places. Floating Point mode is more precise and has higher resolution, but requires a math co-processor — not present on many 386, 486SX, 486SL, 68030, and 68LC040 processors — more memory and more processing time. Fixed Point mode was the native mode of earlier versions of Neuralyst; but now either mode can be selected depending on the circumstances.

The Scaling Margin parameter adds additional headroom, as a percentage of range, to the rescaling computations used by Neuralyst in preparing data for the neural network or interpreting data from the neural network. All Input, Target, and Output data actually processed by the neural network must be in the range 0 to 1. As a consequence, all data presented or interpreted is rescaled from the actual worksheet values to values in the range of 0 to 1. Setting Scaling Margin above 0, increases the amount of headroom or margin, decreasing the range onto which an Input, Target, or Output, is mapped. As an example, if the Scaling Margin is set to 0.1, or 10%, then 10% margin is allowed and split between the high end and the low end of the range, resulting in a mapping to the range 0.05 to 0.95 rather than 0 to 1. This margin can facilitate precision in problems that have continuously variable data or that may exceed the current values in the future by a small amount. Scaling Margin modifies the Min or Max limits that may be set for an input or target by the designated amount.

The options of Input Noise, Force Zero, and Adaptive Learning Rate will typically increase the total number of training epochs it takes to achieve a successfully trained neural network over a “no-frills” approach. However, when used judiciously, the quality of training - and the corresponding quality of predictions — of the neural network can be notably improved. Fixed Point calculation mode generally provides the fastest training; however, for continuously variable inputs or outputs requiring high resolution, Floating Point calculation mode can be more precise. In addition, on high performance Pentiums or PowerPCs, Floating Point approaches Fixed Point speed. Scaling Margin is another control which is useful for problems that have continuously variable inputs or outputs to allow higher precision.

8.3.4 Row Description Block

The screenshot shows a Microsoft Excel spreadsheet titled "LOGIC.XLS" with the following data:

| | P | Q | R | S | T | U | V | W | X |
|----|---|----------------|---|--------------------|---|------------------|---|------------------|---|
| 1 | | | | | | | | | |
| 2 | p. Portions Copyright © 1990-1994 EPIC Systems Corp | | | | | | | | |
| 3 | | | | | | | | | |
| 4 | | | | | | | | | |
| 5 | Row Information | | | Column Information | | | | | |
| 6 | 6 | First Row | | # Input Columns | | # Target Columns | | # Output Columns | |
| 7 | 9 | Last Row | | 2 | | 3 | | 3 | |
| 8 | 4 | Number of Rows | | Input Columns | | Target Columns | | Output Columns | |
| 9 | 1 | Rows/Pattern | | A | | D | | H | |
| 10 | 1 | Row Offset | | B | | E | | I | |
| 11 | | | | A,B | | F | | J | |
| 12 | | | | | | D,E,F | | H,I,J | |
| 13 | Enhanced Parameters | | | | | | | | |
| 14 | Sigmoid Function | | | | | | | | |
| 15 | 1 | Function Gain | | | | | | | |
| 16 | FALSE | Force Zero | | | | | | | |
| 17 | 0 | Zero Threshold | | | | | | | |
| 18 | FALSE | Adaptive LR | | | | | | | |

Figure 8-20 Row & Column Description Blocks

The Row Description Block contains the row information bounding the defined data and the spacing and offset information describing the pattern window.

The First Row and Last Row values mark the beginning and end of the Problem Definition Area, respectively. The Total Number of Rows represents the rows within and including those bounding rows.

Rows/Pattern describes the number of rows that are included in each pattern presented to the neural network. It can be considered as describing the *height* of the pattern window. Row Offset describes the number of rows to shift between patterns presented to the neural network. It can be considered as describing the step size of the pattern window. A Row Offset equal to Rows/Pattern means that the pattern window is stepped the same number of rows as its height, in other

words a series of contiguous but non-overlapping two-dimensional input patterns is defined.

The intersection of the rows set in the Row Description Block and the columns listed in the Column Description Block constitutes the Problem Definition Area.

8.3.5 Column Description Block

The Column Description Block contains the column lists that describe the columns that constitute the defined data and represent each component of the pattern window.

The first three fields of the block, # Input Columns, # Target Columns, and # Output Columns, have counts of the number of columns of each type. Below each count is the actual list of Input Columns, Target Columns, and Output Columns. Each column is listed in its own cell in sequence, but a terminating cell contains all the columns of that type as one string.

The Mode Flag Column? field shows either FALSE, indicating no Mode Flag Column is set, or the column that has been set. If a Mode Flag Column is not set then all rows, as defined by the other parameters, are input to the neural network when either **Train Network** or **Run/Predict with Network** are executed. If a Mode Flag Column is set then the values in the fields are used to switch between training or testing/running modes.

The Min Scale Row? field shows either FALSE, indicating no **MIN scale row is set, or the number of the row that has been set. If no MIN scale row is set, then the neural network takes its minimum value for rescaling by searching each column to find the minimum. If a MIN scale row is set then the values in each column are used as the minimum for rescaling purposes.**

The Max Scale Row? field shows either FALSE, indicating no MAX scale row is set, or the number of the row that has been set. If no MAX scale row is set, then the neural network takes its maximum value for rescaling by searching each column to find the maximum. If a MAX scale row is set then the values in each column are used as the maximum for rescaling purposes.

The Symbol Row? field shows either FALSE, indicating no SYMBOL row is set, or the number of the row that has been set. If no SYMBOL row is set, then the neural network expects that all column data will be numeric only. If a SYMBOL row is set then those columns that have a Symbol List defined should contain symbols that are only from that Symbol List, those columns that do not have a Symbol List defined should contain numeric values only.

The intersection of the rows set in the Row Description Block and the columns listed in the Column Description Block constitutes the defined data and the description of the pattern window.

8.3.6 Network Description Block

| | AB | AC | AD | AE | AF | AG | AH | AI | AJ |
|----|---------------------|--------|-----------------|----------|----------|----------|----------|----------|----------|
| 1 | | | | | | | | | |
| 2 | | | | | | | | | |
| 3 | | | | | | | | | |
| 4 | | | | | | | | | |
| 5 | Network Information | | | | | | | | |
| 6 | # Layers | Valid? | Network Weights | | | | | | |
| 7 | 3 | TRUE | -1.60693 | -1.63518 | 1.459551 | 5.143627 | 5.123119 | -2.40044 | 4.080883 |
| 8 | # Neurons per Layer | | -5.96504 | -1.61473 | 4.870485 | 3.585479 | -1.70692 | -3.54887 | 1.839369 |
| 9 | 2 | | -3.25186 | -0.66611 | 5.503088 | -6.06227 | -2.08621 | 0 | 0 |
| 10 | 3 | | | | | | | | |
| 11 | 3 | | | | | | | | |
| 12 | | | | | | | | | |
| 13 | | | | | | | | | |
| 14 | | | | | | | | | |
| 15 | | | | | | | | | |
| 16 | | | | | | | | | |
| 17 | | | | | | | | | |
| 18 | | | | | | | | | |

Figure 8-21 Network Description & Weights Blocks

The Network Description Block contains information describing the structure of the neural network.

Layers indicates the number of layers in the neural network. Two of the layers are there by default, the input and output layers, and are always part of the count so the minimum value is 2.

Neurons per Layer lists the number of neurons in each layer in sequential rows. The first row of the field is the number of neurons in the input layer, the last row is the number of neurons in the output layer. These two numbers are fixed and defined by the number of Input columns and Target or Output columns. The hidden layers are variable and the number can be set when defining the neural network structure with **Set Network Size**.

There is also a Valid? flag which indicates whether or not the information in the Network Weight Block is correct or may have been changed and needs to be redefined.

8.3.7 Network Weights Block

The Network Weights Block contains a copy of the weights of the neural network and is kept current as the neural network is trained.

When first initialized, the weights are set to random values and there is no meaning to their values. After the neural network has been trained, the value of the weights represent all the learning the neural network has done to that point. Once a neural network has been trained, you should be careful about accidentally re-initializing or otherwise changing these weights; there is no way to reconstruct the weights without retraining the neural network.

It is possible to make a copy of the current weight set and save it elsewhere on the worksheet. Then if you want restore the saved weight set, copy it back to the Network Weight Block. In order for this to work, the network size must not have changed or the network size must be restored to the same configuration as at the time the weight set was saved.

In some cases, it is possible to analyze or interpret the weights. The **Histogram Weights** and **Unpack Weights** commands provide you with some methods of viewing the weights when attempting this.

The **Histogram Weights** command will display the distribution of weight values of the neural network in a histogram chart (see Section 8.2.10).

The **Unpack Weights** command provides a formatted display of the neural network weights. This display is organized by neural network layers and by neurons in each layer (see Section 8.2.11).

8.3.8 Genetic Statistics Block

The screenshot shows a Microsoft Excel spreadsheet with the following data:

| | L | M | N | O | P | Q | R | S | T |
|----|-----------------------------|------------------|---|---|-----------------------------|-------------------|---|---|---|
| 24 | Genetic Training Statistics | | | | Genetic Training Parameters | | | | |
| 25 | 50 | Generation Count | | | 0.75 | Inclusion Rate | | | |
| 26 | 3 | Structure Count | | | 4 | Max Layers | | | |
| 27 | 0.24965 | Least RMS Error | | | 30 | L2 Neuron Limit | | | |
| 28 | 100 | Least Epochs | | | 10 | L3 Neuron Limit | | | |
| 29 | | | | | 0 | L4 Neuron Limit | | | |
| 30 | | | | | 0 | L5 Neuron Limit | | | |
| 31 | | | | | 0.5 | Min Learning Rate | | | |
| 32 | | | | | 1 | Max Momentum | | | |
| 33 | | | | | 0.03 | Max Input Noise | | | |
| 34 | | | | | 3 | Population Size | | | |
| 35 | | | | | Immigrate | Population Mode | | | |
| 36 | | | | | 1 | Crossovers | | | |
| 37 | | | | | 0.1 | Mutation Rate | | | |
| 38 | | | | | Train Error | Fitness Criteria | | | |
| 39 | | | | | 100 | Fitness Limit | | | |
| 40 | | | | | | | | | |
| 41 | | | | | | | | | |

Figure 8-22 Genetic Statistics & Parameters Blocks

The Genetic Statistics Block displays a brief summary of Genetic Supervisor results when it is in operation. There are four fields: Generation Count, Structure Count, Least RMS Error, and Least Epochs. The Structure Count field is updated every time another structure is evaluated. The Generation Count, Least RMS Error, and Least Epoch fields are updated after all the structures of a population are evaluated.

The Generation Count and Structure Count fields maintain numeric counts of the current structure under evaluation. The Generation Count will count up to the generation limit set in **Run Genetic Supervisor**. When this count is reached, genetic optimization will halt. At this time, the best structure may be retrieved, or genetic optimization may be resumed under different Population Mode, Genetic Operator, or Fitness Criteria settings. The Structure Count will count up to the Pool Size set in the Population Mode settings. On evaluating the last structure of a population, the structure that best meets the Fitness Criteria that is set will have its RMS Error and epoch count reported in the Least RMS Error and Least Epochs fields.

8.3.9 Genetic Parameters Block

The Genetic Parameters Block contains fifteen parameters, Inclusion Rate, Max Layers, four Layer *n* Neuron Limits, Learning Rate, Momentum, Input Noise, Pool Size, Pool Mode, Crossovers, Mutation Rate, Fitness Criteria, and Fitness Limit, that bound the limits of the genetic optimization process and change the behavior of the Genetic Supervisor.

The Input column Inclusion Rate controls the average rate of Input column inclusion in structure initialization. The user may select a nominal percentage from 1 to 100% which represents the average rate of inclusion. If the user wants to force all Input columns to be included, then the user can set 100%. The default setting is 75%.

The first and last layer of each neural network is automatically determined by the number of inputs and outputs defined for the original neural network model. Within those constraints, the optimized neural network can have from two to six layers and a wide range of variations in number of neurons per layer. The user can constrain the maximum number of layers and the maximum number of neurons per hidden layer. An implicit rule is that if a lower hidden layer is zero, then higher hidden layer must be zero. The default maximum settings are 4 layers, an input layer, 2 hidden layers, and an output layer; with 30 neurons in the first hidden layer and 10 neurons in the second hidden layer.

There are three network parameters which can be optimized by the Genetic Supervisor: Learning Rate, Momentum, and Input Noise. Learning Rate and Momentum can vary from almost 0 to 1, while the Input Noise can vary from 0 to 0.1. The user can constrain the minimum value of Learning Rate, the maximum value of Momentum, and the maximum value of Input Noise. The default values are 0.5 for Learning Rate, 1 for Momentum, and 0.03 for Input Noise.

There are two controls which determine the management of populations. The first is a control which sets the total population size. Changing this sets the number of structures initialized or evaluated in each generation. The second is a control which chooses among the three modes: Closed Pool, Immigration, or Emigration. With Closed Pool set, then the initial population pool will only be cross-bred or mutated with no new structures initialized. With Immigration set, the population pool will be evolved and every generation new structures will replace the weakest structures of the existing pool. With Emigration set, the population pool will be evolved and every generation the best structure will be emigrated to an entirely new population. The default settings are for population size to be 3 and Immigration to be enabled.

There are two settings the user can control to determine the mechanism for genetic optimization. These settings are Crossover frequency and Mutation Rate. The primary mechanism is cross-breeding controlled by Crossover frequency, where structures are intermingled to create new structures. The crossover rate determines the amount of intermingling. The maximum setting for crossover rate is 10. The secondary mechanism for creating new structures is mutation. With mutation, structures are chosen at random, then features are randomly changed to new values. The mutation rate sets the percentage of structures which will undergo a mutation rather than a cross-breeding to create the new population. The maximum setting for mutation is 100%. Mutation is always in place of cross-breeding; the two never occur at the same time on a single structure. The default settings are for crossover rate to be 1 and mutation rate to be 10%.

The evaluation of each structure is based training or testing while comparing the best RMS Error level achieved or the least number of epochs. There are four modes which can be set: Train Epochs, Train

Error, Test Epochs, and Test Error. Train Error finds the structure with the least RMS Error while holding epochs to the set Fitness Limit. Train Epochs finds the structure with the least epochs to achieve the RMS Error level set in Fitness Limit. These two operate on training data only. Test Error finds the structure with the least RMS Error in the test set data while holding epochs to the Fitness Limit. Test Epochs finds the structure with the least epochs to achieve the RMS Error level of the test set data as set in Fitness Limit. These two measure RMS Error by evaluating the testing data. The default Fitness Criteria mode is Train Error with Fitness Limit set to 100 epochs. This will optimize RMS Error while training each candidate solution to an epoch limit of 100.

8.3.10 Genetic State Block

| | U | V | W | X | Y | Z | AA | AB | AC |
|----|---|---|---|---|---------------------------|---|----|----|----|
| 24 | | | | | Genetic State Information | | | | |
| 25 | | | | | 0.75 | | | | |
| 26 | | | | | 2 | | | | |
| 27 | | | | | 3 | | | | |
| 28 | | | | | 4 | | | | |
| 29 | | | | | 1 | | | | |
| 30 | | | | | 1 | | | | |
| 31 | | | | | 0 | | | | |
| 32 | | | | | 0 | | | | |
| 33 | | | | | 4 | | | | |
| 34 | | | | | 30 | | | | |
| 35 | | | | | 10 | | | | |
| 36 | | | | | 0 | | | | |
| 37 | | | | | 0 | | | | |
| 38 | | | | | 0.5 | | | | |
| 39 | | | | | 1 | | | | |
| 40 | | | | | 0.03 | | | | |
| 41 | | | | | 0 | | | | |

Figure 8-23 Genetic State Block

The Genetic State Block contains information necessary for the Genetic Supervisor to retain intermediate state information. It is not

a complete state information area and so it is not sufficient to support a full reload operation between Neuralyst sessions.

There is no command to support interpretation or display of the genetic state information.