NeuroShell 2 Files - Spreadsheet Format

File Passing

Once you have opened a problem, you can use any of the NeuroShell 2 modules without selecting that problem again. Double clicking on an icon passes the appropriate problem file to each module. The file name appears in the title bar at the top of each module.

Each module is an independent subprogram which you can minimize under Windows. Advanced users may run the subprograms autonomously as long as they know the program arguments. Also, you can have several modules running simultaneously if you know what you are doing. For example, you can train several problems at once. See below for details on the NeuroShell 2 file naming conventions.

NeuroShell 2 File Handling

The NeuroShell 2 **internal file format is either Lotus 1-2-3 .WK1 format or Microsoft Excel .XLS files for Excel releases up to and including Release 4 worksheets.** Users may load the files into their usual spreadsheet and view the data. All major spreadsheet programs as well as almost all database management programs and many other programs are able to work with either the .WK1 or .XLS format, an almost universal format. Users with Excel Release 5.0 or higher can save the file as an Excel Release 4 Worksheet or below. **All references to .XLS files in the manual refer to Excel Release 4 or below worksheets.**

Note: Processing a .WK1 or .XLS file through NeuroShell 2 may remove graph and range name information that is stored in the file. We suggest that you make a copy of the file prior to processing it through NeuroShell 2.

You may have to modify the file depending upon whether or not it contains label information.

1. Files With Label Information

If the file contains label information, the first time you use a .WK1 or .XLS file in NeuroShell 2 it must be imported with the Spreadsheet File Import module so NeuroShell 2 will find out where the label row is and which row is the first data row. (Files may contain more than one row of labels, but only one row may be used for column names. These labels will become NeuroShell 2 column names. The other information rows are not used by NeuroShell 2.) Advanced spreadsheet users can bypass this step by adding their own range information. Refer to the following instructions for details.

a. Using Spreadsheet File Import:

This procedure adds range information to the file that tells NeuroShell 2 which row will be used for column labels and which is the first data row. This module also changes the name of the .WK1 or .XLS file to a .PAT file as the default setting. Refer to File Extensions below for further information.

b. Adding Your Own Range Information

If you don't import your spreadsheet, NeuroShell 2 requires that you define two ranges in your spreadsheet file before it may be used as a NeuroShell 2 internal file:

WK1Label - This range points to the first column in the row that contains label information.

WK1Begin - This range points to the first column in the first row where data begins. This row must be after the label row.

In an Excel spreadsheet, for example, to define the WK1Begin range, you would click the mouse on the cell in the first column in the first row where data begins, then select Define Name from the Formula Menu. When an edit box appears that requests a range name, type

in WK1Begin. Repeat the procedure for cell in the first column in the row which contains label information. Define the name of this range as WK1Label.

If you choose to define your own range information, you must save the spreadsheet in a .WK1 or .XLS format with a .PAT extension so it be used by other NeuroShell 2 modules. (You may also save the file with either a .TRN or .PRO file. Refer to File Extensions in the next section for details.)

Once the range information has been added to a .WK1 or .XLS file, you may add data to the file in your spreadsheet program and then use it in NeuroShell 2 without repeating the import procedure. If you change the row which contains column labels or change the position of the first data row, you must repeat the import procedure or change the range information.

2. Files Without Label Information

If you are working with a file created in a spreadsheet program that does not contain label information, all you have to do is to rename the .WK1 or .XLS file to a .PAT file (or .TRN or .PRO.) If you do use range names and there are no labels, set Wk1Label to the same cell as WK1Begin. You do not need to import this type of file.

Number of Columns and Rows

Even though NeuroShell 2 files are based on the .WK1 or .XLS format, they are not bound by spreadsheet limits on the number of rows and columns. For example, if you import an ASCII file with the **ASCII File Import** module, the file may contain up to 64,000 rows and 16,000 columns.

File Extensions

.A2 Neuron Activations: When applying Backpropagation networks, checking the "write neuron activations to file for slab number" box will cause a file to be created which contains the neuron values for slab number N, the number in the edit box. For example, if you are using the example problem lines and want to write out the neuron values for slab number 2, the file created which contains the neuron values in slab 2 would be called LINES.A2. This facility is for advanced experts only..

.C, C Source Code File: This file contains the C source code for a trained network. This file is created in the Runtime module, Source Code Generator option.

.C01 Contribution Factors: Selecting the Export Contributions to Disk option in the Contribution Factors module will write out a file with the problem name, the contribution factor of each input, and a .CO1 extension into the same directory which contains the problem files.

.DSC Description File: When you open a problem, an edit box is displayed in which you can type a description of the problem. This description is a text file which NeuroShell 2 saves with a .DSC description. If you do not enter any text, a .DSC file is still created for the file which contains the default text.

.FLA, Formula for a Calculator Source Code File: This file contains the formula for a trained network that may be entered into a calculator. This file is created in the Runtime module, Source Code Generator option.

.FIG Configuration File: This file contains the network's architecture, i.e., number of inputs, outputs, hidden layers, connection weights, learning rate, and momentum, etc. The .FIG file is created by the Design module.

.L01 Last Net File: When training ends in Backpropagation networks the very last status of the network is written out in an .L01 (or other appropriate net number. Refer to .N01 files below.) so if you want to continue training from where you left off you will be able to do so. The net used when you apply the trained network will *not* be that net but will be the .N01 etc. net saved for the best training set or test set. The .L01 etc. nets are loaded when you reenter the Backpropagation learning module and you click on the yes button to respond to the prompt "There is already a trained network for this problem. Do you wish to load and continue training?"

.MMX Minimum/Maximum Files: These files contain the minimum and maximum values for the network's input and output variables as well as information on which columns from the spreadsheet or datagrid are to be included in the network. The .MMX file is created by the Define Inputs and Outputs module.

.N01 to .N16 Files, the Network File: This file contains the network's **weights** between neurons and other aspects of the network's "stored knowledge." This file is created in the Learning module and is not readable by the user.

The file number varies from 1 to 16, depending upon the type of network that is being created (in later releases of NeuroShell 2 there will be other numbers). This name difference allows you to experiment with different network types using the same problem name. Once you have created a pattern file in NeuroShell 2, simply select a different network architecture from the Architecture and Parameters module and retrain the network. To apply different networks that you have trained for the same problem, select the network architecture in the Architecture and Parameters module before using the Apply module.

N01 Backpropagation network with each layer connected to the immediately previous layer, 1 hidden layer, and Beginner's System

N02 Backpropagation network with each layer connected to the immediately previous layer, 2 hidden layers

N03 Backpropagation network with each layer connected to the immediately previous layer, 3 hidden layers

N04 Backpropagation network with each layer connected to every previous layer, 1 hidden layer

N05 Backpropagation network with each layer connected to every previous layer, 2 hidden layers

N06 Backpropagation network with each layer connected to every previous layer, 3 hidden layers

N07 Backpropagation recurrent network with feedback from input layer to input layer

N08 Backpropagation recurrent network with feedback from hidden layer to input layer

N09 Backpropagation recurrent network with feedback from output layer to input layer

N10 Kohonen network

N11 Probabilistic Neural Network, PNN

N12 General Regression Neural Network, GRNN

N13 Backpropagation network with 2 slabs in the hidden layer (Ward net)

N14 Backpropagation network with 3 slabs in the hidden layer (Ward net)

N15 Backpropagation network with 2 slabs in the hidden layer and a jump connection between the input and output layers (Ward net)

N16 Group Method of Data Handling, GMDH

.OUT Output File: Once the network is trained and applied to a .PAT, .TRN, .TST, or .PRO data file, an .OUT file is created that contains the network's classifications or predictions. The file is in .WK1 format.

.PAT Pattern File: When using either the Beginner's System or the Design and Test portion of the Advanced System, the main NeuroShell 2 data file is expected to have a .PAT file extension. The file is in .WK1 format.

.PRO Production File: When using the Production portion of the Advanced System, the main NeuroShell 2 data file is expected to have a .PRO extension. The .PRO file format is in .WK1 format.

.RLI/.RLO, the Rules Files: These files contain the If/Then/Else rules created in the Rules module. The .RLI files refer to the rules created in the Prenetwork Rules module; the .RLO files refer to the rules created in the Postnetwork Rules module.

.SYI/.SYO, the Symbol Translate Files: These files contain instructions for converting text into numeric values in the Prenetwork Symbol Translate module and for converting numeric values into text in the Postnetwork Symbol Translate module.

.TRN Training File and .TST Test File: These data files are created by the **Test Set Extract** module. The .PAT file is divided into a .TRN file which is used to train the network and a .TST file which is used with Calibration. (The .PAT file remains intact.) The .TRN and .TST files are in .WK1 format. You can also train on just the .PAT file if you are not using Calibration, or using a paradigm that does not use Calibration, such as Kohonen.

.VB, Visual Basic Source Code File: This file contains the Visual Basic source code for a trained network. This file is created in the Runtime module, Source Code Generator option.

.WTS, the Weight File: This file contains the weight values written out in full precision for all links in the network. This file is created in the Learning module, Show Weights option, for backpropagation and Kohonen networks.