Neural Network Add-in

Version 1.5 Software User's Guide

Contents

Overview

The Neural Network Add-in provides a suite of software tools for developing artificial neural networks (ANN) via a user-friendly interface within Excel. The software implements key steps in the development of ANN models:

- Data pre-processing
- Input variable selection using partial mutual information (PMI)
- Data splitting for model training, testing and validation
- Generalized regression neural network (GRNN) and multi-layer perceptron (MLP) networks

This software user guide is intended to provide instructions on how to implement the features of the software and assumes that the user is familiar with the basic concepts of ANN development. Some references are provided to direct novice ANN practitioners towards further reading, however this guide is NOT intended as a definitive reference on the development or application of artificial neural networks (ANNs).

Getting Started

The software can be installed by downloading and automatically by running the installation program: Neural Network Add-in 1.4 Setup. The installation program will copy all necessary files to a folder (C:\Program Files\Neural Network Add-in) and will automatically register the Microsoft Excel add-in. The Neural Network menu will automatically appear on the Excel Ribbon menu when Excel is next launched.

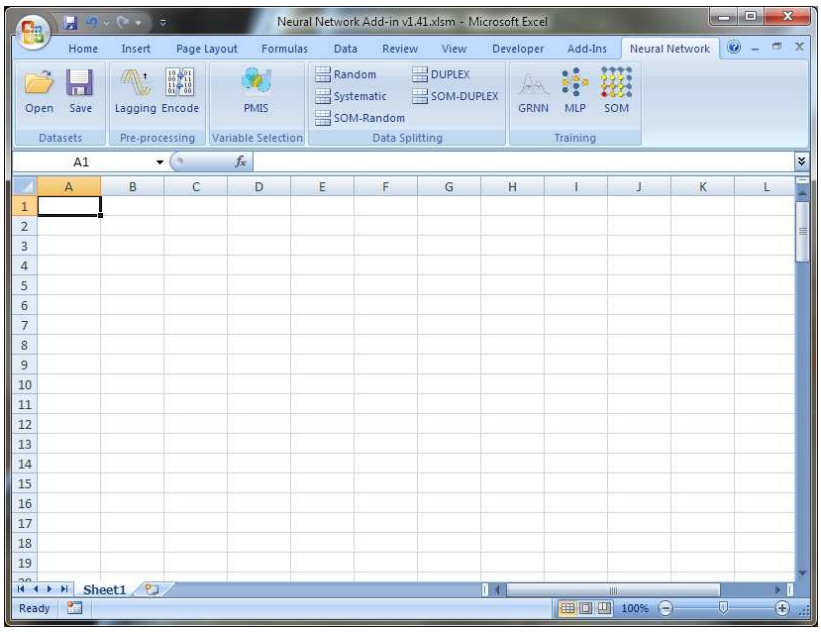
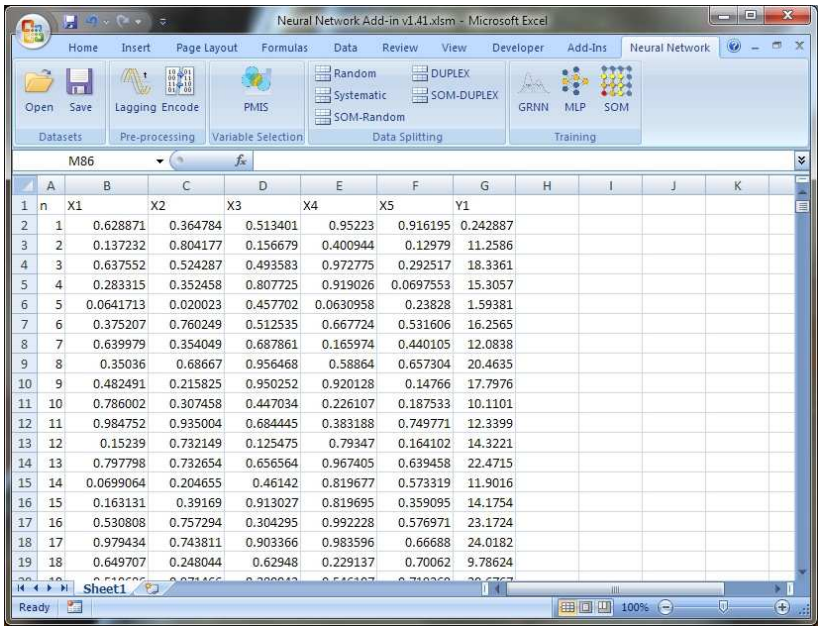


Figure 1 Neural Network Add-in custom menu

Working with Datasets

All algorithms assume a dataset format with columns unique variables, with rows containing corresponding observations of each variable. The data should be labeled, with the first header row specifying the variable labels, and the first column (index) containing labels for each row of data (e.g. time-stamp or number).

Algorithms that are based on analysis of input-output data assume that the final column in any specified dataset corresponds to the output variable. The software does not support any analysis of multiple-input multiple-output (MIMO) datasets.



n	X1	X2	X3	X4	X5	Y1
1	0.628871	0.364784	0.513401	0.95223	0.916195	0.242887
2	0.137232	0.804177	0.156679	0.400944	0.12979	11.2586
3	0.637552	0.524287	0.493583	0.972775	0.292517	18.3361
4	0.283315	0.352458	0.807725	0.919026	0.0697553	15.3057
5	0.0641713	0.020023	0.457702	0.0630958	0.23828	1.59381
6	0.375207	0.760249	0.512535	0.667724	0.531606	16.2565
7	0.639979	0.354049	0.687861	0.165974	0.440105	12.0838
8	0.35036	0.68667	0.956468	0.58864	0.657304	20.4635
9	0.482491	0.215825	0.950252	0.920128	0.14766	17.7976
10	0.786002	0.307458	0.447034	0.226107	0.187533	10.1101
11	0.984752	0.935004	0.684445	0.383188	0.749771	12.3399
12	0.15239	0.732149	0.125475	0.79347	0.164102	14.3221
13	0.797798	0.732654	0.656564	0.967405	0.639458	22.4715
14	0.0699064	0.204655	0.46142	0.819677	0.573319	11.9016
15	0.163131	0.39169	0.913027	0.819695	0.359095	14.1754
16	0.530808	0.757294	0.304295	0.992228	0.576971	23.1724
17	0.979434	0.743811	0.903366	0.983596	0.66688	24.0182
18	0.649707	0.248044	0.62948	0.229137	0.70062	9.78624

Figure 2 Example dataset format

Open a Dataset

Open an existing dataset file via a file browser dialog.

1. Click Open to launch the Open Dataset dialog
2. Browse for a file with a valid tab-delimited(*.txt) or data (*.dat) file-extension
3. Click Open

The selected file will be imported into the active worksheet.

Save a Dataset

Export the active worksheet as a tab-delimited (.txt) file.

To save a dataset:

1. Click Save to launch the Save Dataset dialog
2. Browse for the folder and file, or specify a new filename if the file does not already exist
3. Click Save

The active worksheet will be exported in a tab-delimited text format.

Data Pre-processing

Data pre-processing is commonly employed step prior to analysis during ANN model development. Two commonly used data pre-processing functions are provided in this software:

- Lagging (for time-series data)
- Encoding (scaling/standardizing)

Lagging

Lagging dialog allows the user to transforms single time-series of observations of $x(t)$ into a multivariate dataset, where each variable corresponds to a delay $x(t-d)$, and optionally generate a forecast series of future values $x(t+h)$. To create a lag dataset:

1. Click 'Lagging' to launch the Lagging dialog
2. Select the source data worksheet in the drop-down list
3. Specify the maximum number of delays
4. (*Optional*) Define the lagged variable label suffix (creates lagged variable names with the format <*variable*><*label*><*i*> for each i^{th} delay).
5. (*Optional*) Choose to keep the last column of the unlagged dataset as the final column of the lagged dataset
6. (*Optional*) Choose to generate a forecast series and specify the forecast horizon (look-ahead time-step)
7. Click OK

The output will be generated in a new worksheet that is created in the active workbook, and which will be given the default worksheet name "<*name*>.lagged".

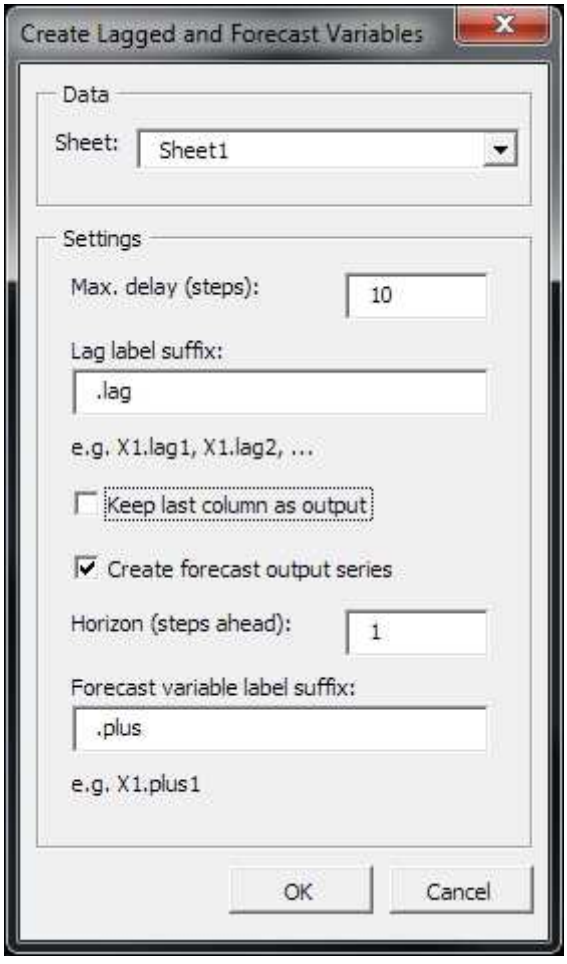


Figure 3 Lagging dialog

Encoding

The Encode dialog provides two alternative encoding schemes:

- Standardisation to zero-mean and unit standard deviation
- Linear scaling transformation from [min, max] to [a,b]

To encode a dataset:

1. Click ‘Encode’ to launch the Encode dialog
2. Select the source data worksheet in the drop-down list
3. Select whether to encode using Scaling or Standardizing transformation
4. If Scaling is selected, specify the minimum and maximum value of the scaled data
5. Click OK

The standardized dataset will be generated in a new worksheet that is created in the active workbook, and which will be given the default worksheet named either “<*name*>.standarsided” or “<name>.scaled”, depending on the selected transformation.

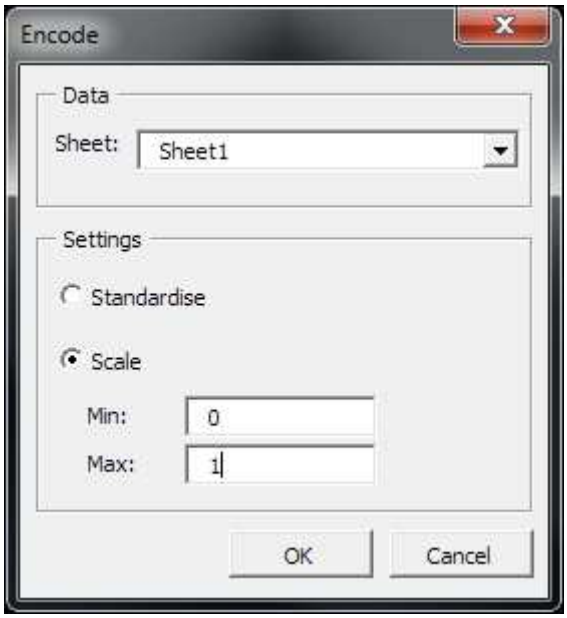


Figure 4 Encode dialog

Input Variable Selection

Input variable selection is used to identify the best set of variables to use as inputs for an ANN model. The goal of input selection is to select a set of variables with maximum predictive power, and minimum redundancy.

Partial Mutual Information Selection

The software provides an implementation of a step-wise selection scheme based on analysis of partial mutual information (PMI). The algorithm iteratively selects variables by first calculating the PMI of each variable, and selecting the one that maximises the PMI.

To select input variables using PMI-based variable selection:

1. Click ‘PMIS’ to launch the PMI selection dialog
2. Select the source data worksheet in the drop-down list

3. (*Optional*) Select to stop the analysis a defined number of iterations and specify the maximum number of iterations (Note: b*y default, the program will iterate until all candidate input variables have been analysed in order of importance, or you can optionally terminate selection at a specified number of iterations*)
4. (*Optional*) Choose whether to use a bootstrap analysis to estimate critical values of mutual information and specify the bootstrap size
5. Click OK.

The program will generate output in a new worksheet that is added to the active workbook, which is named "<name>.pmi".

The output of the PMI analysis is a summary table, indicating the variable selected at each iteration, the corresponding value of PMI. The remaining columns correspond to a number of other criteria, which can be used to determine the optimal number of input variables.

The different criteria are interpreted as follows:

- I^*(BS-95) is the 95th percentile upper-bound for a bootstrap estimate of the critical value of the mutual information, which is the noise threshold for estimating MI based on a finite sample of data. If the PMI for a variable is higher than this value (i.e. $I'(x;y) > I^*$), then the variable should be included as an input. I^*(BS-99) is the corresponding 99th percentile upper-bound, which can alternatively be used in a similar manner.
- I^*(MC-95) is the 95th percentile upper-bound of the critical value of the mutual information that has been pre-determined for Gaussian *i.i.d.* bivariate data. If the PMI for a variable is higher than this value (i.e. $I'(x;y) > I^*$), then the variable should be included as an input. I^*(MC-99) is the 99th percentile upper-bound, , which can alternatively be used in a similar manner.
- AIC(k) is the Akaike Information Criterion (AIC), which a measure of the trade-off between model complexity and the information within the set of inputs, as a function of the number of inputs, k. The iteration corresponding to minimum AIC(k) represents the optimum number of inputs to be selected. This criterion uses a penalty term $k+1$ to determine the AIC. AIC(p) is the AIC(k) with the penalty determined for a kernel regression, which is used to determine partial mutual information (PMI).
- The Hampel score, Z, is an outlier test applied to the variable with the highest PMI value, relative to the distribution of PMI values for all variables. Variables with Z value greater than three (i.e. $Z > 3$) should be selected.

Starting with the first iteration, the criteria are applied to determine the relevance of a candidate input variable. If the variable is relevant, the variable should be included and the next candidate variable considered. If the variable is not relevant, the variable should not be included, and selection should be terminated (no more inputs should be selected). Only one criterion should be applied consistently for all variables to make selections.

Bootstrap analysis provides a noise threshold for the value of mutual information that is dependent on the sample size. A large bootstrap size will increase the accuracy and reliability of the associated stopping criteria, but will also significantly increase the computational run-time of the analysis.

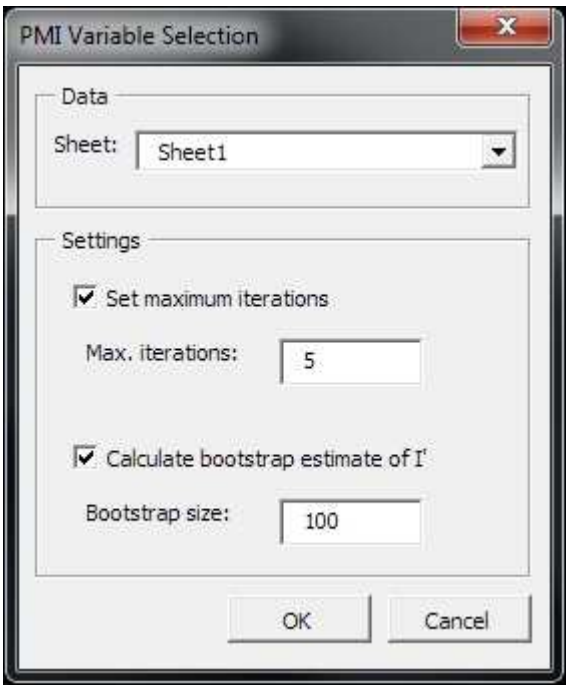


Figure 5 PMI variable selection dialog

Data Splitting

Data splitting is used during ANN model development to generate separate, independent datasets for training, testing and validating ANN models. The software supports several alternative algorithms for data splitting.

Random Data Splitting

The random data splitting implements a data split using uniform random sampling to generate training, testing and validating datasets.

To generate a random data split:

1. Select Random from the data splitting algorithm drop-down menu
2. Select the source data worksheet in the drop-down list
3. Specify the proportions (%) for training, testing and validating data
4. Check to initialize the random number generator with CPU time or a specific seed value
5. (*Optional*) Specify the seed value for the random number generator
6. Click OK

The program will generate three new worksheets in the active workbook, which will be named "<name>.train", "<name>.test", and "<name>.valid", corresponding to training, testing and validating data samples, respectively.

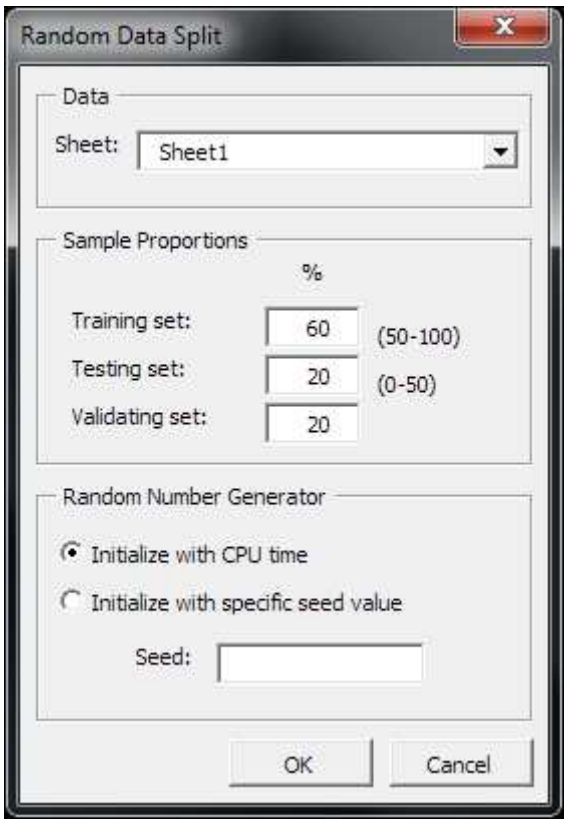


Figure 6 Random data splitting dialog

Systematic Stratified Data Splitting

Systematic stratified sampling implements data splitting using systematic data sampling (sampling each m^{th} datum) of a dataset that has been sorted on the output variable (last column of the dataset). The combination of sorting and systematic sampling effectively evenly samples data according to a stratification of the output variable, which can ensure that data representative of all output conditions are selected for a sample. The resulting data split will vary according to the randomly selected starting point on an interval [1, m].

To perform systematic data splitting:

1. Select Systematic from the data splitting algorithm drop-down menu
2. Select the source data worksheet in the drop-down list
3. Specify the proportions (%) for training, testing and validating data
4. Click OK

The program will generate three new worksheets in the active workbook, which will be named "<name>.train", "<name>.test", and "<name>.valid", corresponding to training, testing and validating data samples, respectively.

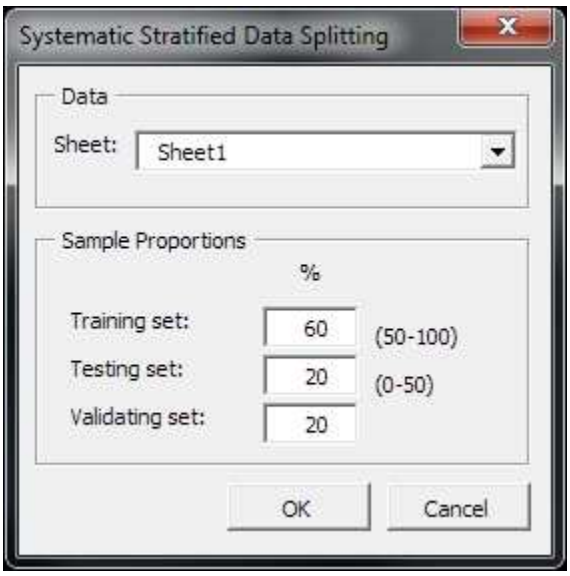


Figure 7 Systematic stratified data splitting dialog

SOM-based Stratified Data Splitting

Random stratified sampling is based on random sampling of data from a stratified or clustered dataset. Stratification delineates regions within a dataset for which data within the same region are similar, but distinct from data in other regions. Stratified random sampling is intended to improve on uniform random sampling applied over the data by reducing the overall variability of sampling and increasing the representativeness of data.

This software implements multivariate stratified random sampling using the self-organising map (SOM) to perform multivariate clustering, and intra-cluster sampling based on uniform random sampling.

To perform SOM-based stratified sampling:

1. Select 'SOM-based' from the data splitting algorithm drop-down list
2. Select the source data worksheet in the drop-down list
3. In the Sample tab,
    a. specify the proportions (%) for training, testing and validating data
    b. Specify the sample allocation rule that determines how many data are selected for training and testing (the remaining points are allocated to validating data).
4. In the Training tab,
    a. Specify the size of the SOM grid (length and width)
    b. Specify the learning algorithm parameters (number of training epochs, learning rate and neighbourhood size) for the tuning and ordering phases
5. In the PRNG tab, choose the
6. Click OK

The program will generate three new worksheets in the active workbook, which will be named "<name>.train", "<name>.test", and "<name>.valid", corresponding to training, testing and validating data samples, respectively.

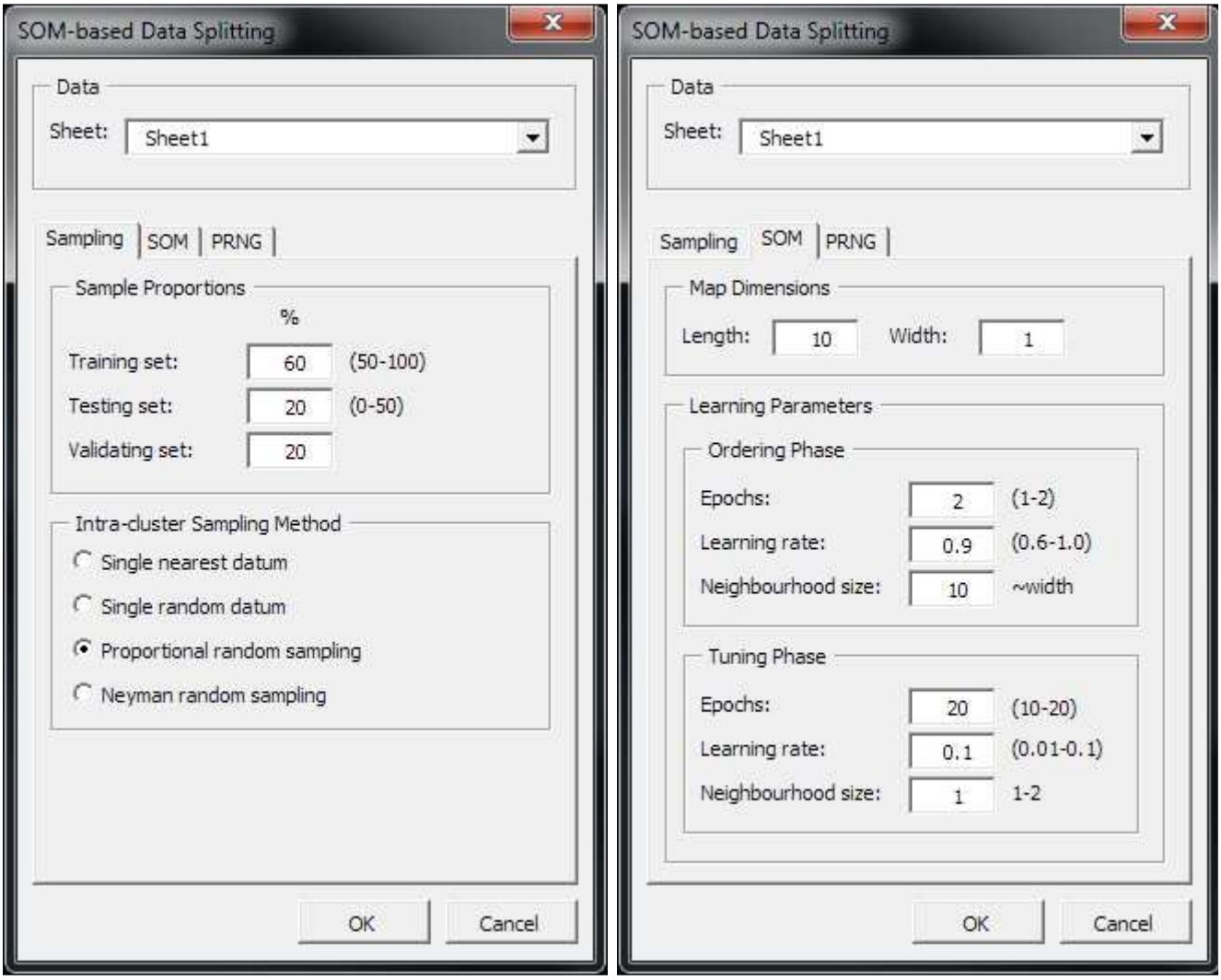


Figure 8 SOM-based data splitting dialog sampling tab (left) and SOM algorithm tab (right)

The most appropriate SOM settings will depend on the data and type of sampling that is subsequently applied to the clusters. The SOM map size is the most influential parameter. A rule-of-thumb is to set the number of map units equal to $\sim N^{0.5}$ and a ratio of approximately 1.6:1 for the dimensions of the map. Typical value ranges for each learning parameter are indicated adjacent the textbox to guide the user on appropriate settings.

DUPLEX

The DUPLEX algorithm is a variation of the Kennard-Stone sampling algorithm. The Kennard-Stone algorithm selects new data into a sample that maximize the distance to data within the current sample. In DUPLEX, data are selected in a pair-wise manner. Since DUPLEX is a fully deterministic algorithm, it will generates only one split for a given dataset, regardless of the ordering of the dataset.

To perform DUPLEX data splitting:

1. Select 'DUPLEX' from the data splitting algorithm drop-down list
2. Select the source data worksheet in the drop-down list
3. Specify the proportions (%) for training, testing and validating data
4. Click OK

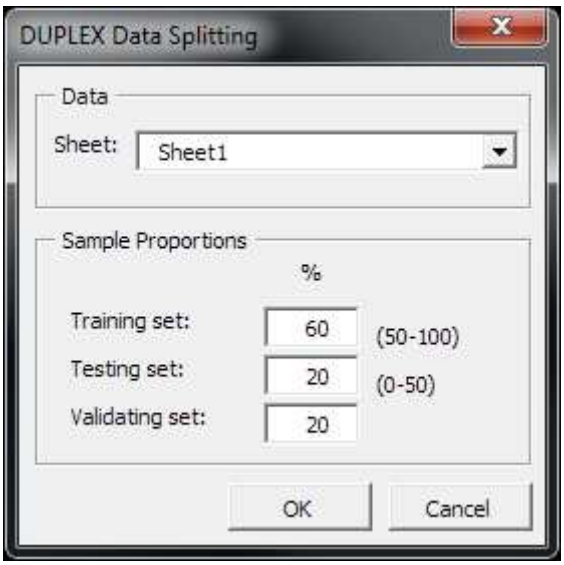


Figure 9 DUPLEX data splitting dialog

The program will generate three new worksheets in the active workbook, which will be named "<name>.train", "<name>.test", and "<name>.valid", corresponding to training, testing and validating data samples, respectively.

SOMPLEX

SOMPLEX is a two-stage sampling technique that performs SOM-based clustering, and then applies the DUPLEX algorithm for intra-cluster sampling. SOMPLEX can provide more consistent results than stratified random sampling, by applying deterministic sampling within clusters. Alternatively, the SOM clustering can effectively improve the computational performance of DUPLEX, since the computational requirement on a clustered database is reduced from $O\{N^2\}$ to $O\{kN_H^{.2}\}$, where $N_H << N$.

To perform SOMPLEX data splitting:

1. Select 'SOMPLEX from the data splitting algorithm drop-down list
2. Select the source data worksheet in the drop-down list
3. In the Sample tab, specify the proportions (%) for training, testing and validating data
4. In the Training tab,
    a. Specify the size of the SOM grid (length and width)
    b. Specify the SOM learning algorithm parameters for the tuning and ordering phases
5. In the PRNG tab, choose the seed number for the pseudo-random number generator
6. Click OK

The program will generate three new worksheets in the active workbook, which will be named "<name>.train", "<name>.test", and "<name>.valid", corresponding to training, testing and validating data samples, respectively.

SOMPLEX may produce some variation in results due to the random initialization of the SOM weights and subsequent variation in the clustering. However, a coarser clustering of the data can be applied by using a smaller map size than for stratified random sampling, which can reduce the variability of clustering. Best results will depend on the specific data, however a rule-of-thumb for setting the map size for SOMPLEX is a total number of map units equal to $\sim(1/5)N^{0.5}$ and a ratio of 1.6:1 for the map dimensions. Typical value

ranges for each learning parameter are indicated adjacent the textbox to guide the user on appropriate settings.

Training

The software provides the capability to train and evaluate the performance of two types of feed-forward neural networks:

- Multi-layered perceptron (MLP)
- Generalised Regression Neural Network (GRNN)

Multi-layered perceptron (MLP)

The multi-layer perceptron (MLP) is the most conventional neural network architecture, and is used in 90% of engineering modeling applications. This software implements a three-layered MLP architecture: input, hidden and output layers. The output layer is restricted to a single output node, but supports a variable number of hidden-layer nodes. The input layer will vary according to the dimensionality of the data.

The training algorithm is based on conventional backpropagation algorithm (BPA) with momentum. Cross-validation against test data is used to implement early-stopping, which avoids over-fitting and ensures good generalization of the trained network.

To train an MLP network:

1. Open the MLP dialog
2. Specify the training, testing and validating data worksheets from the respective drop-down lists (*Note: a warning will be generated if the same data is used more than once*).
3. In the Network tab,
    a. Specify the number of hidden layer neurons (must be greater than or equal to 1)
    b. Specify the transfer function used in the hidden layer neurons
4. In the Output tab,
    a. (*Optional*) Select to generate a of performance evaluation summary worksheet
    b. (*Optional*) Select to generate a worksheet of predicted training, testing or validating data
5. Click OK.

The program will generate output in new worksheets that are added to the active workbook.

Scaling of the data is automatically performed by the program, depending on the type of transfer function that is specified by the user. This ensures the data are correctly scaled so as to avoid saturating the neurons, which impairs the training process.

Generalised Regression Neural Network (GRNN)

The generalized regression neural network (GRNN) is a class of probabilistic neural network (PNN) that provides an ANN representation of kernel regression. The underlying principle is the estimation of the conditional expectation $E(y|x)$ using a kernel-density estimate of the conditional probability density function, $p(y|x)$. The architecture of the GRNN is fixed and does not need to be optimized, which can make them faster to develop than MLP models. However, they are most suitable for smooth functions or surfaces. In comparison to the MLP, training of the GRNN is fast, since training of a GRNN requires the optimization of only the kernel bandwidth parameter. This software uses a fast GRNN training based on Brent's algorithm to determine the optimum bandwidth in several iterations.

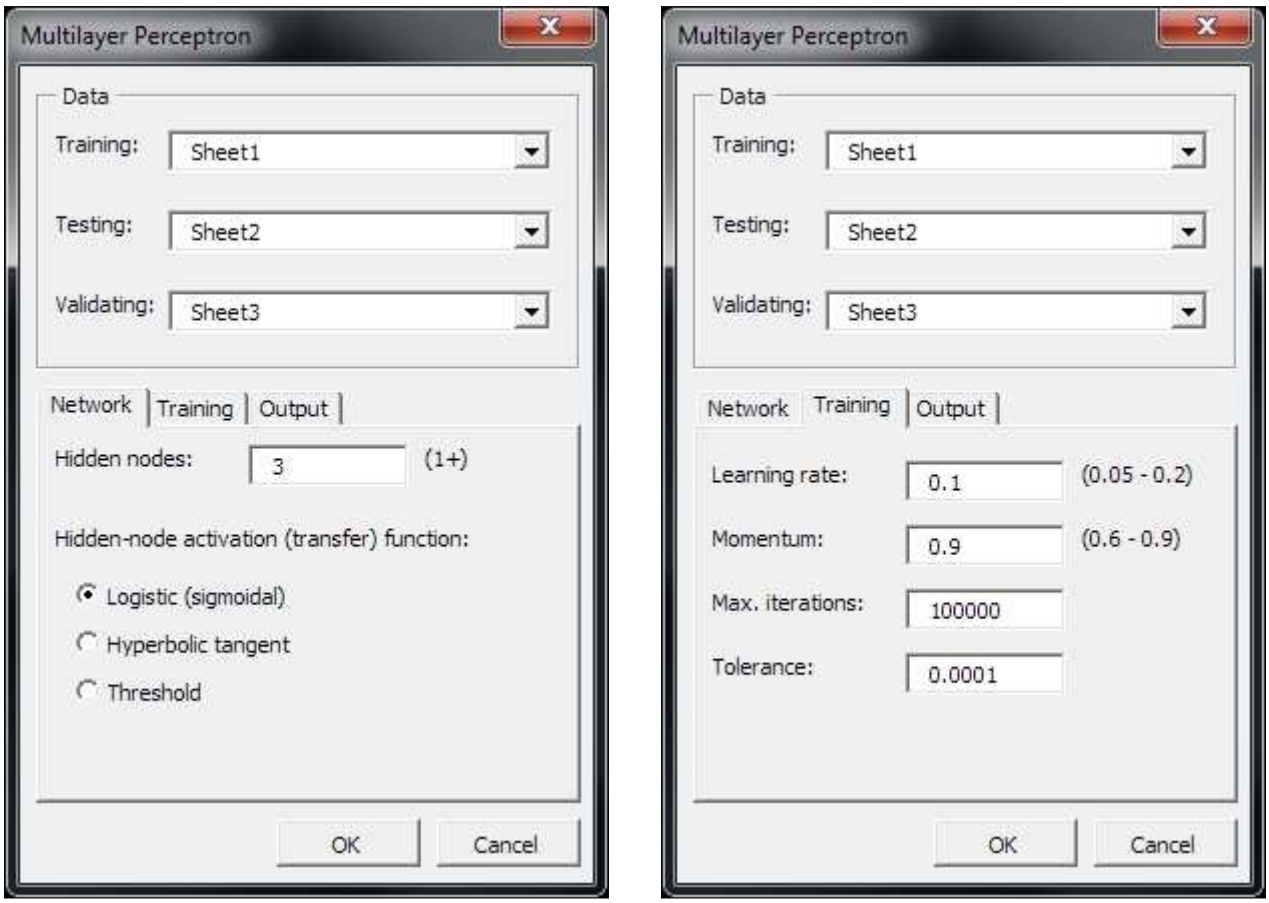


Figure 10 Multilayer perceptron dialog network tab (left) and training algorithm tab (right)

To train a GRNN:

1. Open the GRNN dialog
2. Specify the training, testing and validating data worksheets from the drop-down lists (*Note: a warning will be generated if the same data is used more than once*).
3. Select to train the network, or specify the kernel bandwidth
4. (*Optional*) Select to generate a of performance evaluation summary worksheet
5. (*Optional*) Select to generate a worksheet of predicted training, testing or validating data
6. Click OK.

The program will generate output in new worksheets that are added to the active workbook.

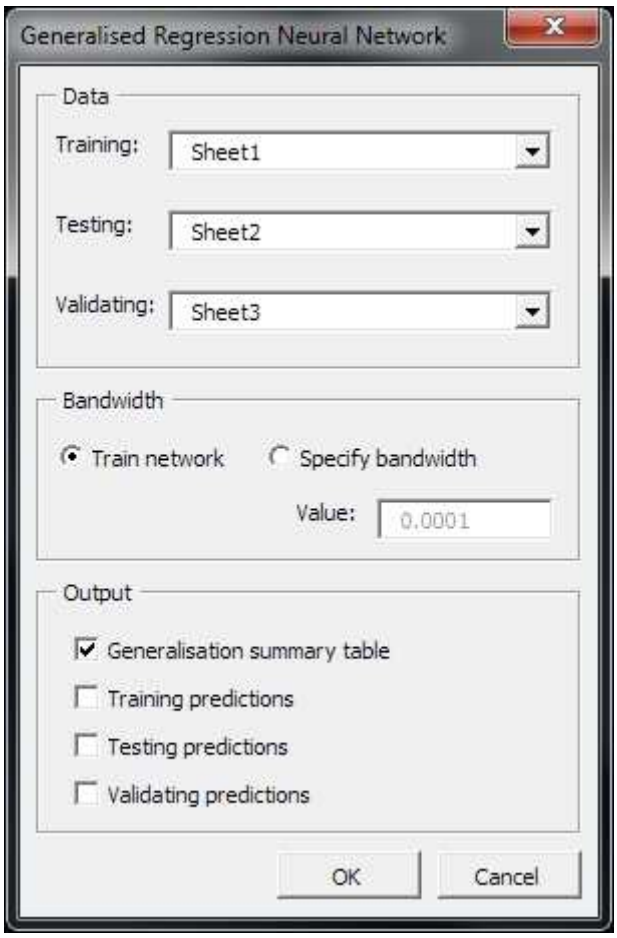


Figure 11 GRNN training dialog

Self-organizing Map (SOM)

The self-organizing map (SOM) is an unsupervised ANN, which implements competitive learning. The SOM maps a set of *d*-dimensional data to an *m* x *n* array (2-dimensional map) of prototype vectors, while maintaining spatial relationships from *d*-dimensional data within the 2-dimensional map. The SOM can be used to cluster multi-dimensional data for data visualization exploration, classification or vector quantization applications.

To train a self-organizing map:

1. Open the 'SOM' dialog
2. In the Data tab, select the training data worksheet in the drop-down list
3. In the Network tab, specify the size of the SOM grid (length and width) *(Note: few universal rules for setting the grid size exist, but one rule for setting the grid size is to use a number of SOM units equal to $k\sqrt{n}$ where k = 1/5 (small), 2(medium) or 5 (large); and use an ratio m:n ~ 1.6)*
4. In the Training tab,
    a. Specify the learning algorithm parameters (number of training epochs, learning rate and neighbourhood size) for the tuning and ordering phases
5. In the Output tab, choose to output
    a. SOM histogram (number of data clustered to each SOM unit)
    b. SOM codebook (list of vectors corresponding to SOM weights)
6. Click OK

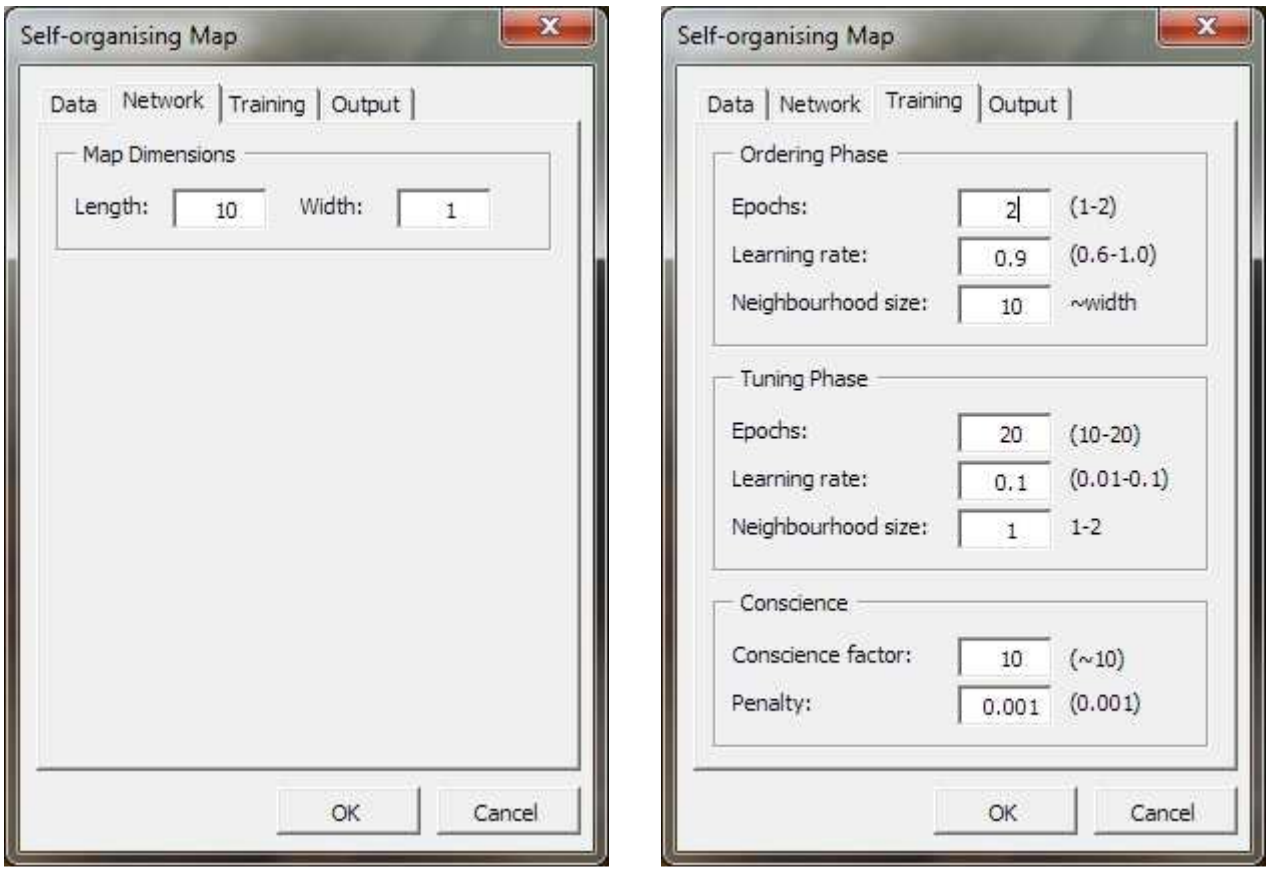


Figure 12 Self-organising map dialog network (left) and training algorithm tab (right)