MONMLP 4.0 –A large overview

Scope and reason of existence:

This software was created to train Artificial Neural Networks (ANN) in order to "learn" a complex function **Y**=**f**(**X**) using **not only** some data pairs (**X**,**Y**), as the most training algorithms (BFGS, Back-Propagation etc) does, but also *a priori* information (when available) on the monotonicity and concavity of the function. (Retain that even if Y is a vector (as X is also) only for one of its components (say Y_1) we can use the monotonicity-and concavity information). The training of the network is performed with a genetic algorithm - genetic hill climber optimizer that determines the networks parameters (weights) by minimizing MSE on all the outputs of the network on a training data set plus some penalty terms accounting for agreement with apriori monotonicity – concavity information that we have for one of the outputs.

An Example in what is useful this software and how to use it is the following:
Suppose we want to learn the function $z=X_1^2+5X_2$ with $X_{1,2} \in [1,10]$. It is depicted in fig 1.

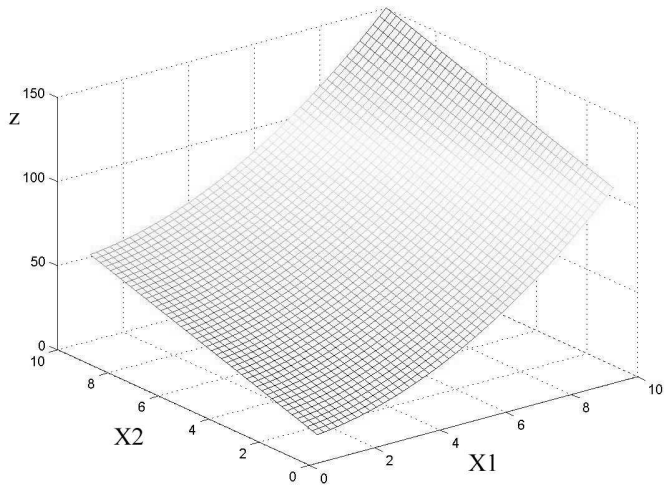


Fig. 1 The original function to be learned.

As is the case in all practical problems we dispose of a reduced set (here 30) of pairs (**X**,y) , where $y=z+\varepsilon$, (ε is a random variable called noise). See Fig 2.

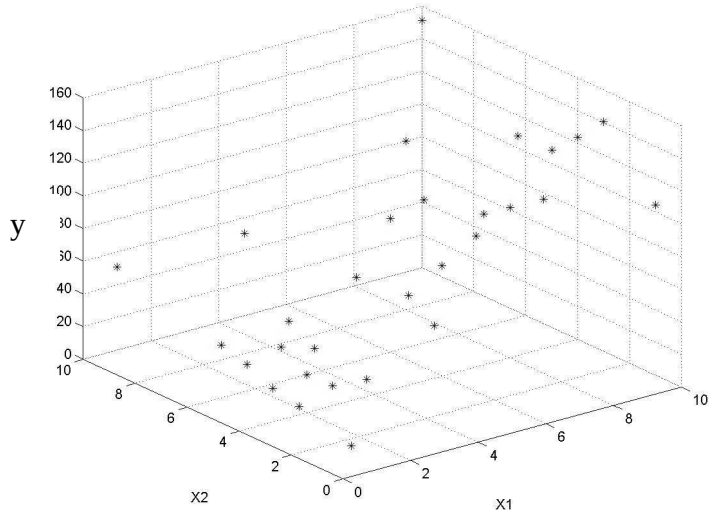


Fig. 2 The noisy training data set.

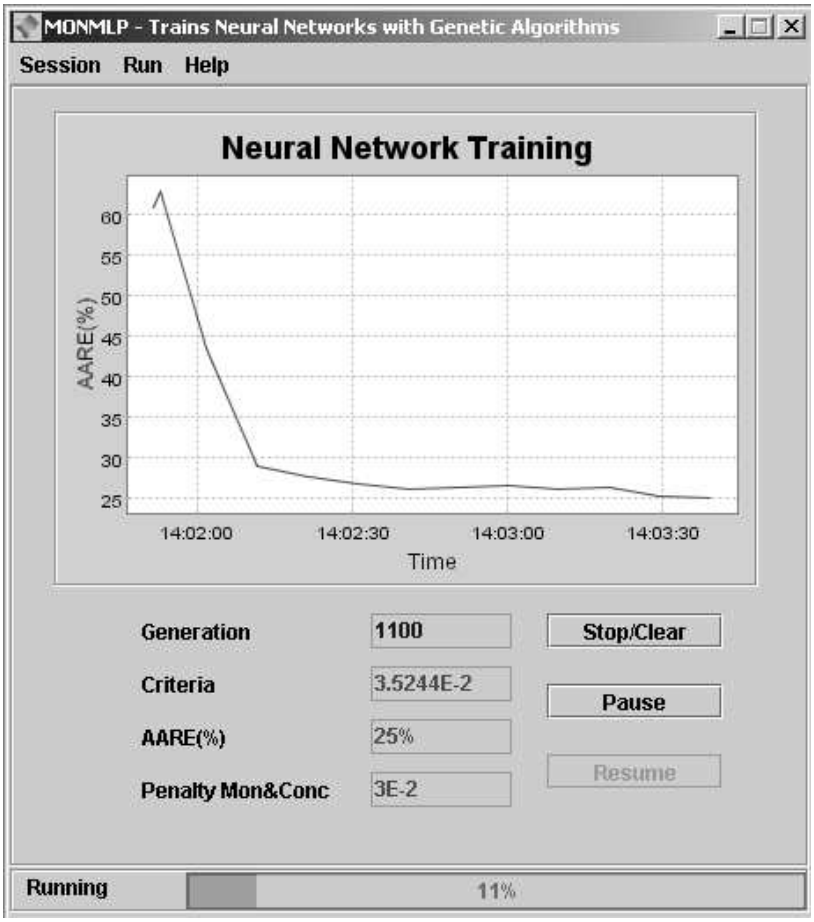
The training file *ex1data.dat* (in MONMLP data format, see appendix for details) is in the *tutorial* directory containing 30 pairs (**X**,y). The first two columns are X_1, and X_2 that will constitute the inputs of the network and the third column is y, which the output.

The supplementary information that we shall use apart from the experimental data samples is monotonicity and concavity information:

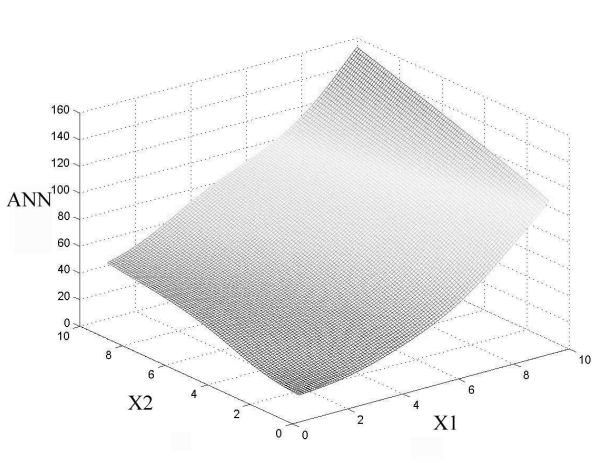
$$\frac{\partial z}{\partial X_1} \geq 0\,,\ \frac{\partial z}{\partial X_2} \geq 0\,,\frac{\partial^2 z}{\partial X_1^2} \geq 0$$

How To Use MONMLP:

1. Execute the Monmlp.bat file and from the *Session* menu choose *Open existing session* and select, from the *tutorial* directory, *ex1conf.cfg* (it is a configuration file edited for exemple). For details on what means each option *Settings* window see appendices.
2. Save now your configuration in whatever location you like (do not save over *ex1conf.cfg* this file, you might need it later).
3. The *Run* button in the *Settings* window being enabled, you may press it to start the training. Watch in the appearing window how the Average Absolute Relative Error (on the training data set) diminishing during the iterations (see Fig. Bellow).

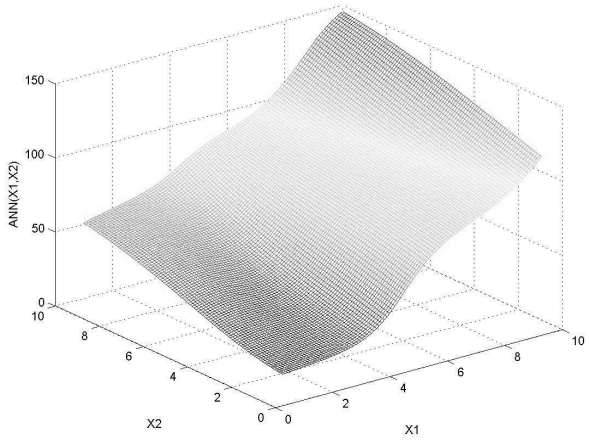


4. After the training is completed, look in the *tutorial* directory to get the resulting files. The file *.ite* contain a record of the training process, *.prt* file contain the target versus ANN predicted values on the training data. The *.w* files contain the network configuration and parameters at different stages of the training process. The number a *.w* file name starts with represents the generation at which the network parameters were saved.
5. The ANN.xls file shows you the training file and the performances of the ANN model built with MONMOLP against one obtained with a BFGS algorithm. The performance is evaluated o a test data set, whose **X** values were never used in the training process.



a)

AARE	**4.87%**
STDEV	**4.30%**



b)

AARE	**7.21%**
STDEV	**8.24%**

Fig. 3 ANN functions obtained with MONMLP a) and BFGS b) training algorithms

As you might see a more accurate model is found if MONMLP is used instead of classic training algorithms.

Appendix

MONMLP data file format:
This is a text file whose columns are Tab separated and each row is in a new line. The columns contain the input and the output data, each row corresponding to a particular observation. From left to right lays the input followed by output columns: In_1 In_2 …In_n Out_1, Out_2

How to configure a new training session

In Fig. 4 lays a screen shoot of the *Settings* window of the program.
Training data file may be selected pressing the *Browse* button, and should contain at least 2 the input-output pairs (two rows). It must be in the *MONMLP data file format* and the inputs as well as the outputs values must be inside the ranges they are in the *all data* file.
All data file (in the *MONMLP data file format*) may be selected pressing the *Browse* button, and should contain at least 2 the input-output pairs (two rows). This file is needed only to extract the ranges for the inputs and the outputs of the network and the inputs and outputs order must be the same as in *the training data file*.

Results path is the location were the resulting files will be saved.

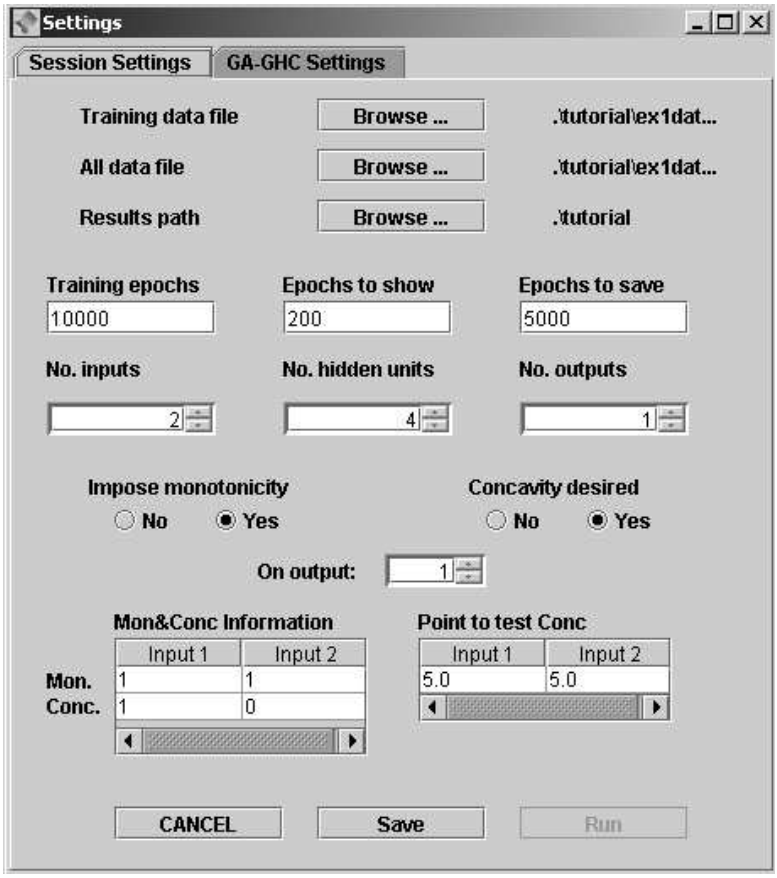


Fig. 4 The *Settings* window, *Session settings* panel.

Training epochs is the number of iterations to perform the training. Usually 50 000 iterations are enough to obtain an accurate model for most problems. However you should try less or more iterations depending if your problem is easy or difficult respectively.

Epochs to show allows specifying the number of iterations after which information like AARE, etc. should be updated on your screen.

Epochs to save permits customizing the frequency of *.w* (weigth) and *.prt* (parity) files saving. Don't set a small value because lot of time will be spent with saving non-useful files as long as only the last files may interest you.

No. inputs sets the number of inputs of the network, which corresponds to the dimensionality of the input vector X.

No. hidden units defines the number of units in the network's hidden layer. You should try several in the vicinity of twice the number of inputs in order to get the smaller model that still predicts well.

No. outputs specifies the number of outputs of the network, that corresponds to the dimensionality of the output vector Y. You may have multiple outputs but on a single one you may impose constraints on monotonicity and or concavity.

On output indicates on which of the outputs the monotonicity-concavity information will apply.

Impose monotonicity decides whether or not monotonicty-concavity information will be specified.

Impose concavity decides whether or not concavity information will be specified.

Mon&Conc information table allows you to describe the monotonicity-concavity information you have. A "1" in the first row of the table in the column "i" stands for $\frac{\partial Y_k}{\partial X_i} \geq 0$ where k is the value specified in the *On output* field, while a "-1" in the first row of the table in the column "i" stands for $\frac{\partial Y_k}{\partial X_i} \leq 0$. A "0" provides no first gradient information for that input.

A "1" in the second row of the table in the column "i" stands for $\frac{\partial^2 Y_k}{\partial X_i^2} \geq 0$ where k is the value specified in the *On output* field, while a "-1" stands for $\frac{\partial^2 Y_k}{\partial X_i^2} \leq 0$. A "0" provides no first second gradient information for that input.

Point to test conc specifies in which point the concavity information should be tested. The values entered here for X must be inside the ranges they are in the *all data* file.

The *GA-GHC settings* control panel should be unchanged for most of the problems.