trainings, the box below the file name should be checked to ensure that the file is read every time. The names (x, f_1 and f_2) and locations (column 1, 2 and 3) for input and output variables are filled in the table. The number of header lines (i.e. lines to be ignored in the beginning of the file) is also specified.

The button named Scaling opens a form where scale factors for the pattern data can be specified, see Figure 5. In the example pattern file, the value of f_2 is in the interval [-1,0]. Since we want to use the sigmoidal activation function (1) for the nodes in the output layer, we specify a factor -1 which transforms f_2 to the interval [0,1]. (Also real values may be used as scale factors.)

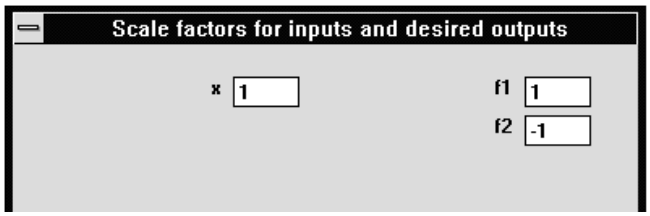


Figure 5.

- Test file... is used to specify an optional file with test patterns. The test patterns are not used for training but the network is evaluated with the test patterns after each iteration, which may indicate whether *overtraining* occurs [8,9]. The locations (columns) of the variables in the test file must match those in the pattern file and all data pre-treatment parameters specified for the training (see Setup|Network...) are used also for the test data. In this present example, no test file is used.

- Output file... specifies an optional file which holds data created by feeding the training patterns to the network after training. The output file is normally used when further analysis or graphical presentation of the results is needed. In the present example, we choose to write the pattern index and the activations of the hidden nodes to the output file, see Figure 6.

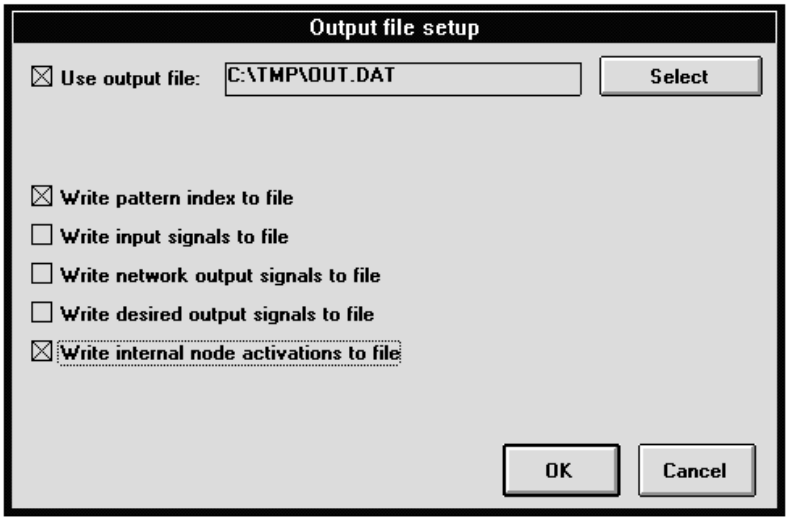


Figure 6.

- Log file... specifies an optional file which, after each iteration, is updated with time (from the computer's clock), iteration number, SSQ, rms error and the Marquardt parameter (λ). If a test file is used, the rms error for the test set may also be written to the log file. The log file is especially suited for non-interactive runs, while the log table (see Show menu) is available interactively.
- Network type specifies the network algorithm, so far only MLP networks are available.
- Network... opens the MLP setup window, shown in Figure 7, which is the main form for specifying network configuration and training options.

In the frame for training specifications, the first boxes are used for selecting pre-treatment of the data in the pattern (and test) file. A sliding-window mean value filter is available and the size of the training data can be reduced by specifying the number of lines to be read from the pattern file before each new pattern is formed. In the present example, the data pre-treatment is switched off. The next box in the frame specifies the total number of (filtered) patterns to be formed from the pattern file data. In the example, only the first 95 patterns in the file are used. The optimisation task switch is used for specification of the open parameters in the training. As alternatives to the network weights, gain terms for the input and/or output signals can be used. The maximum number of iterations in training is given in the last box in the frame, but the default value (1000) is normally more than enough.

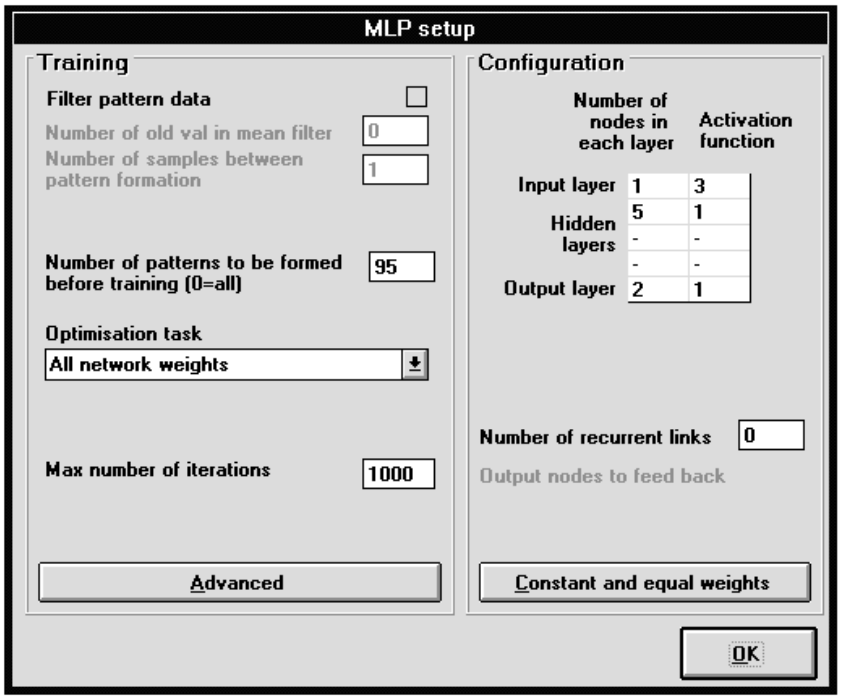


Figure 7.

The configuration frame holds specifications of the network. The number of hidden layers and nodes in each layer are specified and the activation functions for hidden and output nodes are selected. The numbers of input and output nodes are also shown, but they can only be changed in the pattern file setup window. In the example, one hidden layer with five nodes is used. The sigmoid (1) is used as activation function for both hidden and output nodes. The last two items are used for recurrent networks only, see example II.

The Advanced button in the MLP setup opens a form for selection of more specialised training options, see Figure 8.

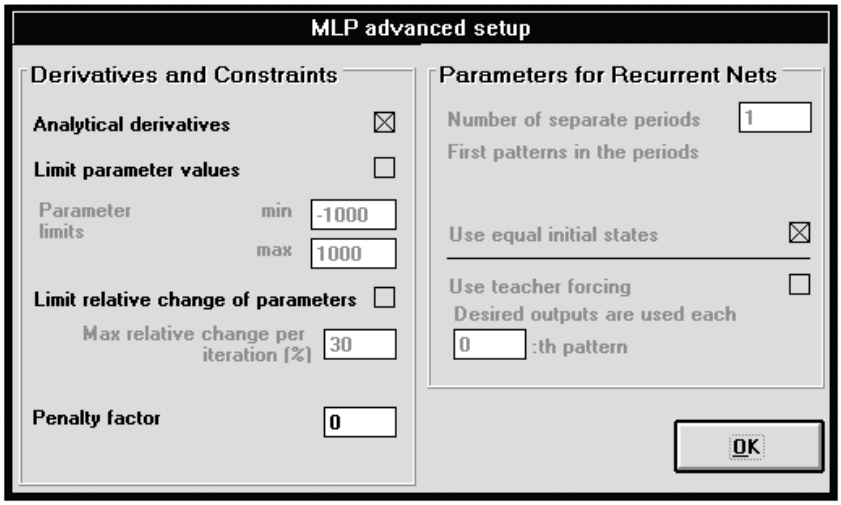


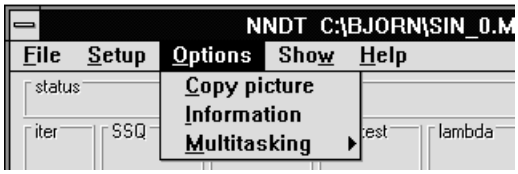
Figure 8.

The user can choose whether analytical expressions or a numerical estimation shall be used for the derivatives (Jacobian). Analytical derivatives

are used by default since they normally lead to faster convergence. The values of the parameters, i.e. the weights and biases (and initial states for recurrent networks) can be kept between an upper and a lower limit. Also, a maximum relative change of each parameter per iteration step can be specified. Another method to restrict the parameters in the search is to specify a penalty factor (Equation (9)), the use of which will be shown later in this example. As a first approach for the example problem, no weight constraints are specified. The parameters for recurrent networks are discussed in connection with example II.

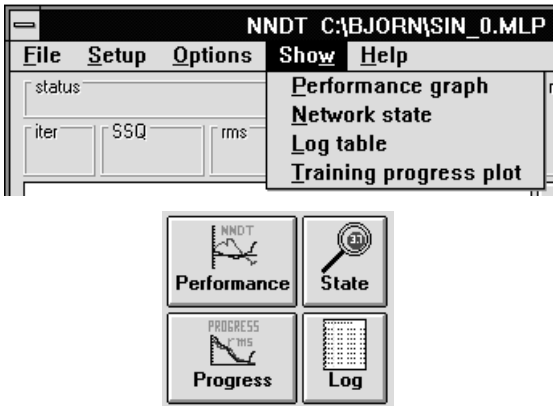
The Constant and equal weights button in the MLP setup displays a form where the user can specify weight equalities and constant weights. This option is used in example II.

Options Menu



- Copy picture produces a bitmap of the network picture and copies it to the Windows environment clipboard.
- Information shows a box with the names of setup file, pattern file, test file, output file and log file and displays the amount of free memory available in the system.
- The Multitasking switch sets the grade of multitasking, i.e. how often NNDT checks if other events are in the system message queue during training. Normally, full multitasking should be used. Limiting the grade of multitasking makes the training a little faster but slows down the system.

Show Menu



The show menu is used to access windows which, in contrast to the setup windows, may be kept open during an entire run. As alternatives to the menu items, the buttons on the main window can be used.

- Performance graph shows a graph in which the plot variables are specified by the user. The performance graph is illustrated in connection with the discussion of the training of the example problems.

- Network state displays a form for examination of network weights and node activations, see Figure 9. The Network weights table lists the elements of the weight vector, cf. Equation (6). Single weights are changed by editing the values in the table. As an alternative to the methods for limiting the weight values specified in the MLP advanced setup, such manual modification can be used to activate paralysed nodes or to force nodes to operate in desired ranges. The activations (y) of all network nodes and the target values, for a user-specified pattern number, are shown in the Network activations table.

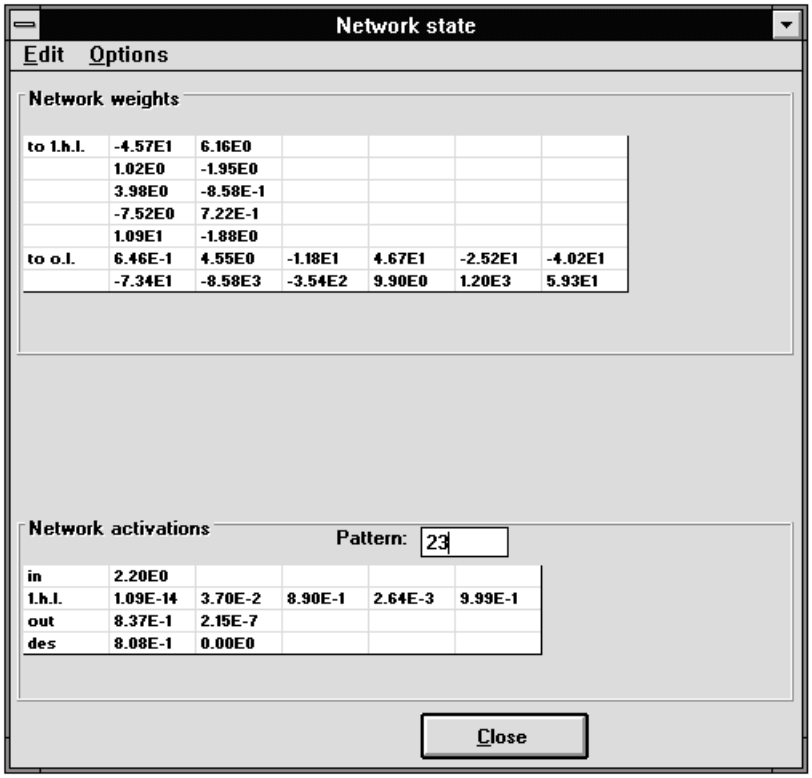


Figure 9.

By selecting Options|Initialise parameters in the network state window, a form for generating uniformly distributed weights, biases and initial states is displayed, see Figure 10. The parameters are initialised by random numbers in the range $[-\alpha,\alpha]$, where α is specified by the user. A separate initialisation range can be given for each layer. In the example, α=0.1 is used for both layers of weights. The seed for the random number generator is given by the user.

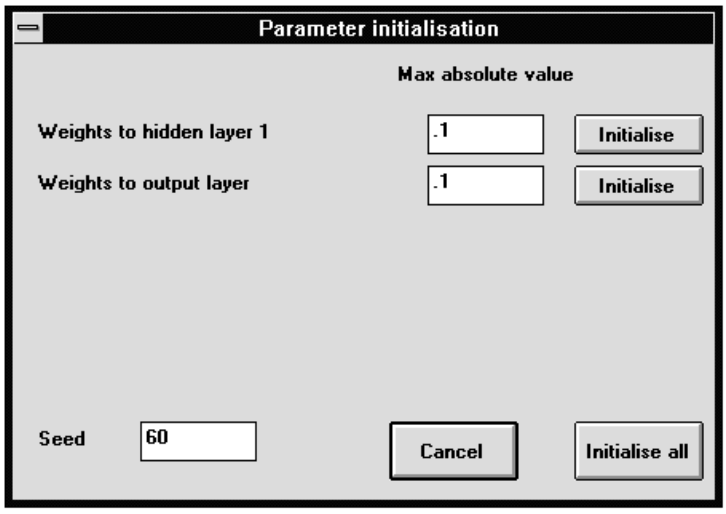
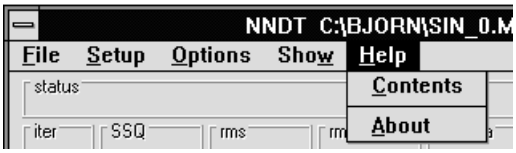


Figure 10.

The Options menu in the network state form also has items for selecting data to be displayed in the network activations table. Train data activations is the default, Test data activations can be used if a test file is selected.

- Log table shows a list which is updated after each iteration with time (from the computer's clock), iteration number, SSQ, rms error and the Marquardt parameter (λ). If a test file is used, the rms error for the test set is also written to the list.
- Training progress plot shows a graphical presentation of the rms error for the training data vs. iteration number. If a test file is used, the rms error for the test set is also plotted.

Help Menu



- Contents starts the system help tool with the index page of the NNDT help file. A faster way to access the help file is to press F1 in any NNDT window; this action automatically shows the associated help page. The help is available also during training.
- About shows a box with information on program author, current version, etc.

Training the Network

When all setup parameters are given, the training is started by simply pressing the "Start" button. Information on the training progress is displayed in the main window. For the present example problem, the training aborts after 40 iterations and a message is displayed telling that no further improvement is possible. Both the error measure (rms = 0.37) and the performance graph shows that the result is far from satisfying. An analysis of the network shows that two of the weights have very large values (Plot|Weights in the performance graph, see Figure 11) and that most of the nodes operate in inactive, i.e. saturated, ranges. To overcome the problem, the weights could be reinitialised with a different seed and the training started again, or, one could modify the large weights in the network state window and continue the training.

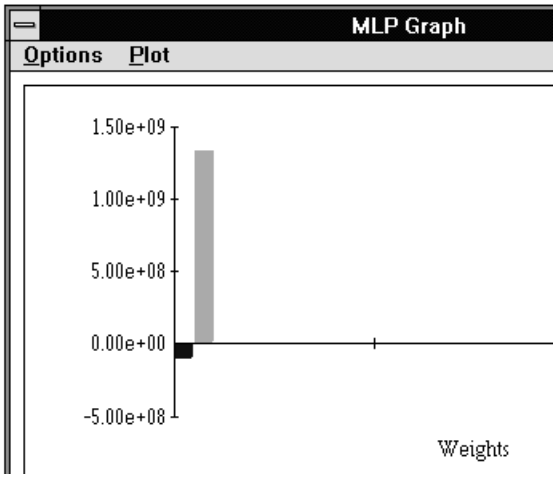


Figure 11.

Another alternative is to use the options for restricting the weight values, which will be shown here. First, the weights are initialised using the same seed (60) to enable comparison with the previous result. In the MLP advanced setup window, a penalty factor of 0.01 is specified, which should prevent very large weight values. The training is started and after 56 iterations the algorithm stops at an rms error of 0.17. The plot of network outputs and desired output signals shows that the network has learned much of the input-output mapping but still not all of it. A plot of the internal node activations vs. pattern index (Plot|Special in the

performance graph, see Figure 12) shows that each hidden node work in its active range in some region of the training data. However, all nodes are saturated at patterns 75 to 95.

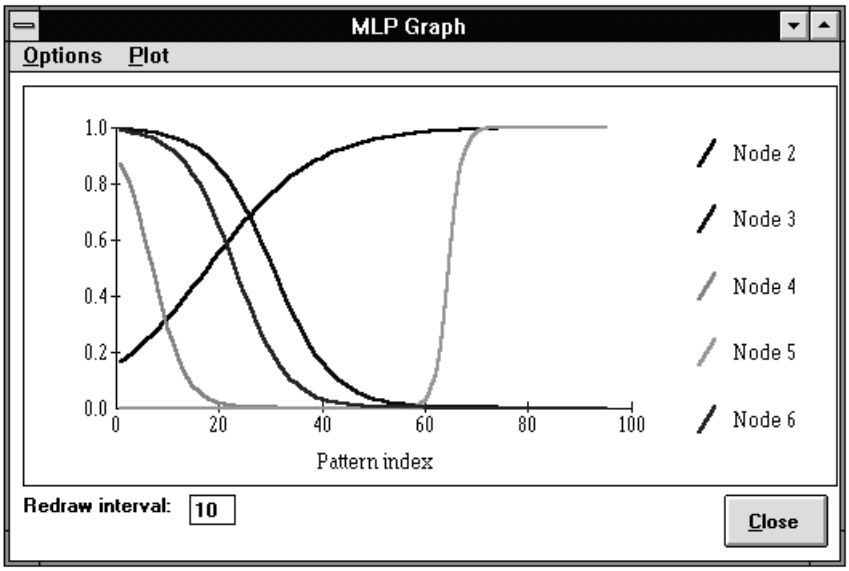


Figure 12.

The penalty term is relaxed (γ=0) and the training is continued. After 100 iterations, the training is stopped manually since it is clearly seen that the input-output mapping is explained well by the network. The rms error is 0.028 and the network performance is good, as indicated in Figure 13. Note that the sign of f_2 in the pattern file is changed by premultiplication by a factor -1, selected in the scale factors window accessed from the pattern file setup form.

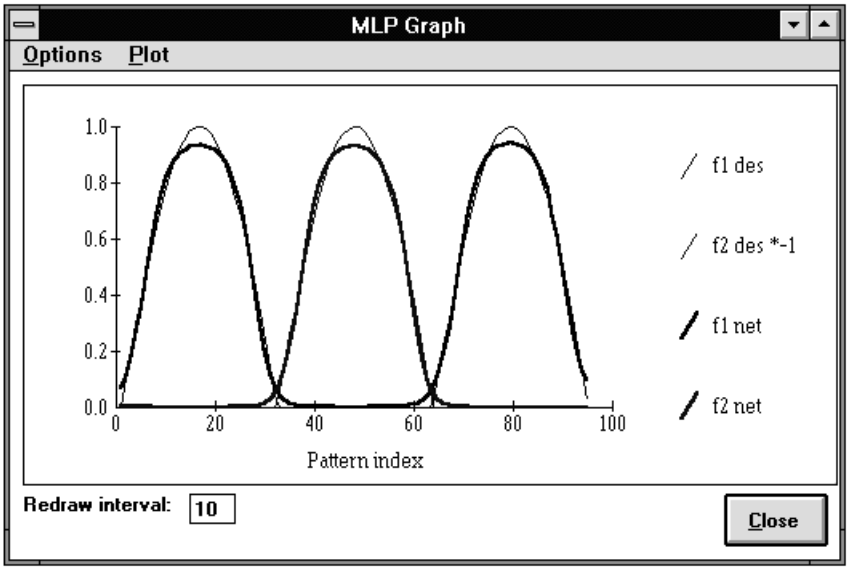


Figure 13.

A plot of the progress in terms of rms error for the last training period (after resetting the penalty term) shows a step-shaped convergence curve, typical for the training algorithm, cf. Figure 14.

Figure 14.