NN-Tool 2024

Edition: October 2023
Author: Frank Baermann

Content

1. Introduction

NN-Tool consists of a simulator for creating neural network models and of an application module for interpreting the created models (prediction and optimization respectively).
After provision of data the models can be created very quickly, because NN-Tool takes over independently the scaling transformations as well as the identification of complete data sets, which are necessary for neural nets (NN). The created NNs are of the standard type (feedforward net with one hiddenlayer), which has been proved to be optimal in all practically relevant task requirements[1]. After creating a model, the user has the possibility to execute evaluation- and optimization calculations on the basis of the created model. The application consists of the three main steps:

- Data preprocessing
- Network training
- Net evaluation for prediction and optimization purposes

In this handbook, the use of NN-Tool is discussed on the basis of three examples. The corresponding example data are to be found in the Excel files **TEST.xls, CLASS6.xls, DataRec_Demo.xlsx** and **ZEIT.xls** which are stored in the folder **"Documents\NN-Tool"**. The folder will be created during the first start of NN-Tool.

[1] Recently these experimental results have been underpinned also theoretically by the work of Barron. However, as training algorithm not backpropagation is used, but due to performance reasons an algorithm is used, which is described in detail in the following publications:
Bärmann, Biegler-König: On a Class of Efficient Learning Algorithms for Neural Networks, Neural Networks **5** (1992) pp. 139-144
Martino, Fanelli, Protasi: Computational Experiences of New Direct Methods for the On-line Training of MLP-Networks with Binary Outputs
ICANN 93 - Sorrento, pp. 627-630

2. Installation

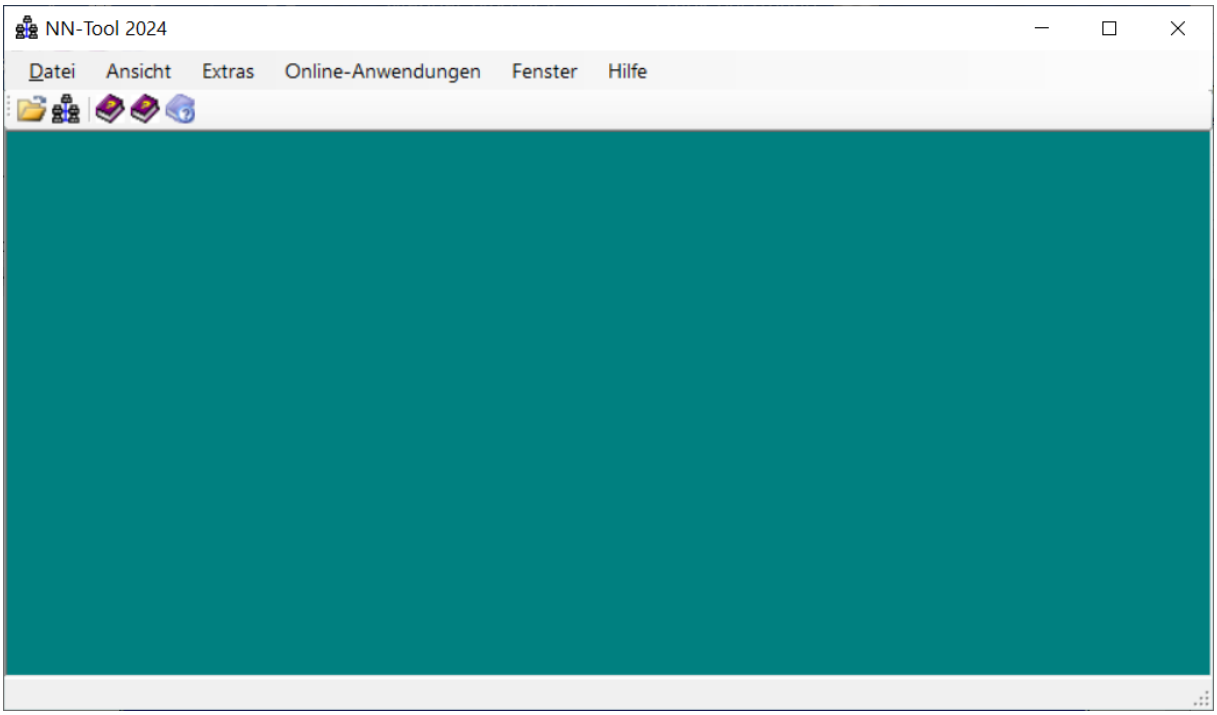
Please note the instructions on the enclosed supplement.

Insert the installation CD and start the program "Setup.exe". Follow the instructions of the installation program.

In case of **installation problems** please consult the last pages of the manual or look in installation errors in NN-Tool Help.

Very Important: Before using NN-Tool please make sure that your number format in the system control / Regional and Language Options / is using the decimal separator which belongs to the **respective area schema**, i.e. in case of area schema **German** the **decimal comma**, in case of area schema **English** (Great Britain) and **English** (USA) the **decimal point**. These settings are standard settings with Windows installations! **Without these settings NN-Tool cannot and will not run correctly!** By the way this applies to a lot of applications in the technological sector. If you are forced to use an alternative decimal separator, you should **always switch your operating system to the corresponding regional and language schema**.

Now please start NN-Tool. The following window might appear (NN-Tool main window, German user interface):



NN-Tool might start in German language mode. Don't Panic! Choose Extras / Optionen and select "Englisch" for "Sprache" (language), then press "Save" and "Exit".

3. Data Preprocessing

NN-Tool requires that the data are in a file with the extension pat (for pattern). Each record has to be in one line, different parameters have to be separated by semicolon. The first record consists of the parameter names. As decimal separator you have to use as a standard the decimal separator which belongs to the respective regional schema (windows standard).

Basically NN-Tool provides two different options to import data or to create a correct data file (Patfile):

- **Direct access to an Excel folder:** instead of loading a Pat-file NN-Tool initially loads an Excel-Application, the user selects a worksheet (with several sheets in the Excel folder) and NN-Tool creates and imports the corresponding data file.

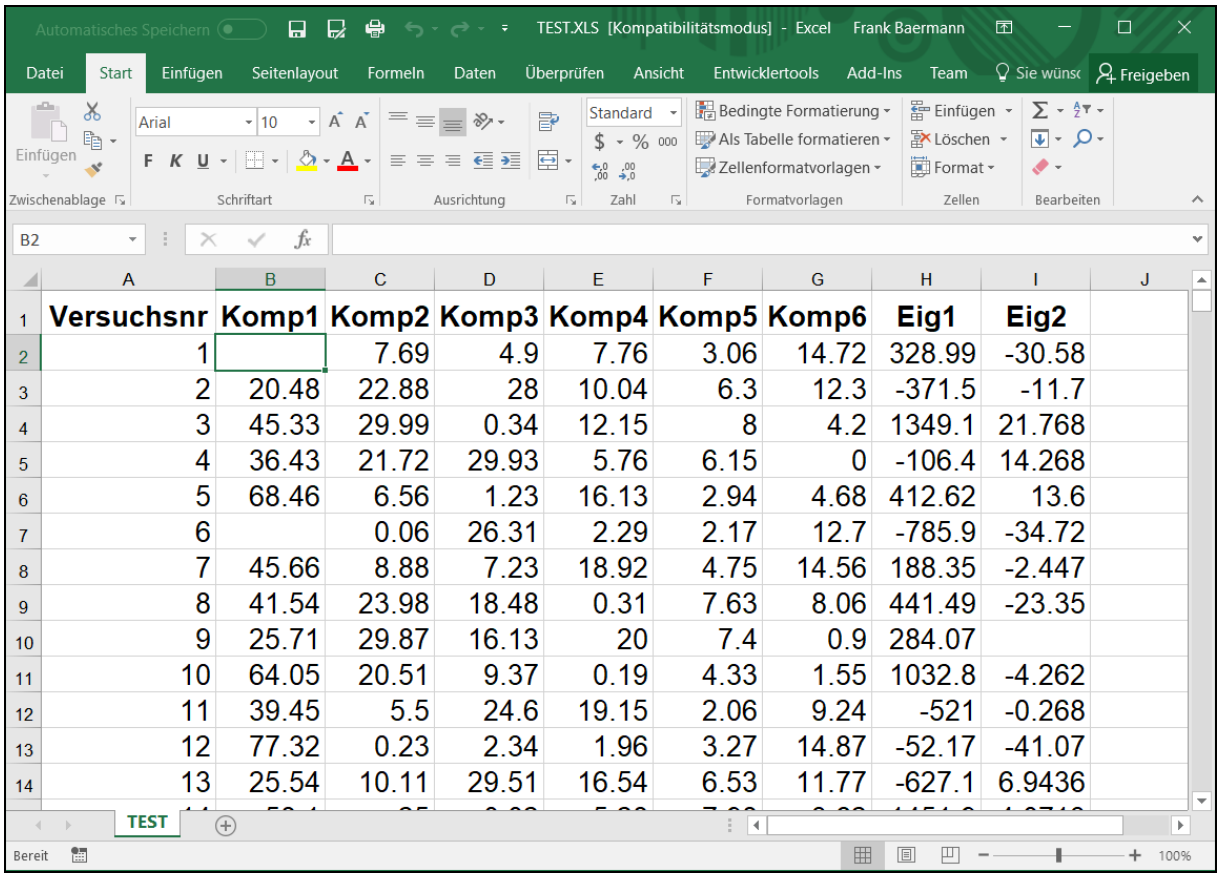
- **Loading a .csv-file**: The so-called .csv-format (**character separated values**) is a widely used file format for data sheets. .csv-files can be created from a large variety of applications (e.g. Excel, Access, etc.).

The methods are introduced in the following. It is assumed that we have got the data in an Excel-Sheet:

1. Direct access to an Excel folder

The simplest method to produce a correct data file is to load the data into an Excel spreadsheet at first. For this you can use the Excel mechanisms for data import. Following that NN-Tool allows you to directly access the sheet and generate the corresponding NN-Tool file.

Now start Excel and open the file **Test.xls** in the folder **"Documents\NN-Tool".** If you can't find that folder, please copy the folder **"NN-Tool"** from the NN-Tool installation directory. The path of the installation directory can be found under **"Extras" / "Options".**
The Excel file looks as follows:



	A	B	C	D	E	F	G	H	I	J
1	Versuchsnr	Komp1	Komp2	Komp3	Komp4	Komp5	Komp6	Eig1	Eig2	
2	1		7.69	4.9	7.76	3.06	14.72	328.99	-30.58	
3	2	20.48	22.88	28	10.04	6.3	12.3	-371.5	-11.7	
4	3	45.33	29.99	0.34	12.15	8	4.2	1349.1	21.768	
5	4	36.43	21.72	29.93	5.76	6.15	0	-106.4	14.268	
6	5	68.46	6.56	1.23	16.13	2.94	4.68	412.62	13.6	
7	6		0.06	26.31	2.29	2.17	12.7	-785.9	-34.72	
8	7	45.66	8.88	7.23	18.92	4.75	14.56	188.35	-2.447	
9	8	41.54	23.98	18.48	0.31	7.63	8.06	441.49	-23.35	
10	9	25.71	29.87	16.13	20	7.4	0.9	284.07		
11	10	64.05	20.51	9.37	0.19	4.33	1.55	1032.8	-4.262	
12	11	39.45	5.5	24.6	19.15	2.06	9.24	-521	-0.268	
13	12	77.32	0.23	2.34	1.96	3.27	14.87	-52.17	-41.07	
14	13	25.54	10.11	29.51	16.54	6.53	11.77	-627.1	6.9436	

The file **TEST.xls** describes a recipe property task. Two physical properties (Eig1 and Eig2) have been measured on 500 recipes, each consisting of 6 components (from Komp1 to Komp6). Additionally, a serial test number ("Versuchsnr") has been assigned as first parameter.
On the basis of this example the application of NN-Tool is explained in the following.

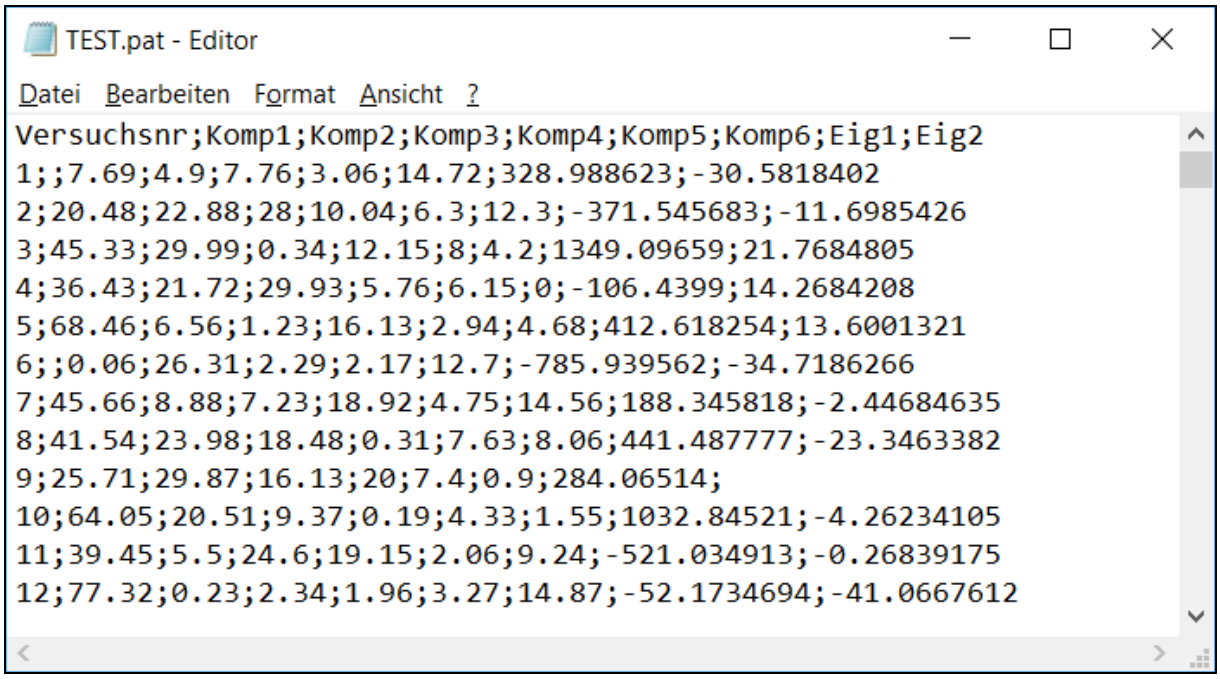
NN-Tool handles an application (i.e. a neural network with all included items: data, weights etc.) under an application name. **All assigned files get the application name as file name**. The only difference between them is their extension. At the beginning of an application only the data file **<application name>.pat** has to exist or it must be generated from Excel or another application (Access, MatLab…).

As you can see in column Komp1, the data is partially incomplete. NN-Tool is able to process incomplete records appropriately, provided that the corresponding values are **really empty** and do not contain blanks or tabulators, etc. NN-Tool interprets blanks and all other non-numerical values as so-called **classifiers**. Therefore, these are not permitted to appear in numerical parameters (if necessary, remove them with an editor). Classifiers are parameters, which exactly take one value (instance) from a finite choice, e.g. the classifier "surface imperfection" with the instances "none", "scratches", "scores", "dents", etc. The second example goes into the processing of classifiers.

Now close the Excel workbook, return to NN-Tool (or start NN-Tool) and proceed as follows:

- In NN-Tool select **File/Load Datafile** or press the appropriate button (the button has the explanatory text "Load Datafile").
- In the **"Open"** dialogue set the file type to **"Excel files (*. xls, *. xlsx)".**
- Select the Excel workbook with the data. Note that the data in Excel must have the same structure as in the above example, i.e. the parameter names in the first row of the Excel spreadsheet, followed immediately by the data records (records arranged in rows, no empty rows). If the Excel folder contains only one data sheet, it will be converted into the Patfile and immediately imported in NN-Tool. **The sheet's name (not the workbook name!) will be used as the name of the Patfile.**
- If the Excel folder contains several data sheets, a hint follows and a selection window is displayed where you can select the right sheet. The first 20 lines of each of the different sheets are displayed to provide additional information.
- Select the sheet you like and press **"Load Sheet"**. A .Patfile is generated and imported from the sheet (of course with all the records, not only the first 20). The data set is displayed in the window „Data Sheet".

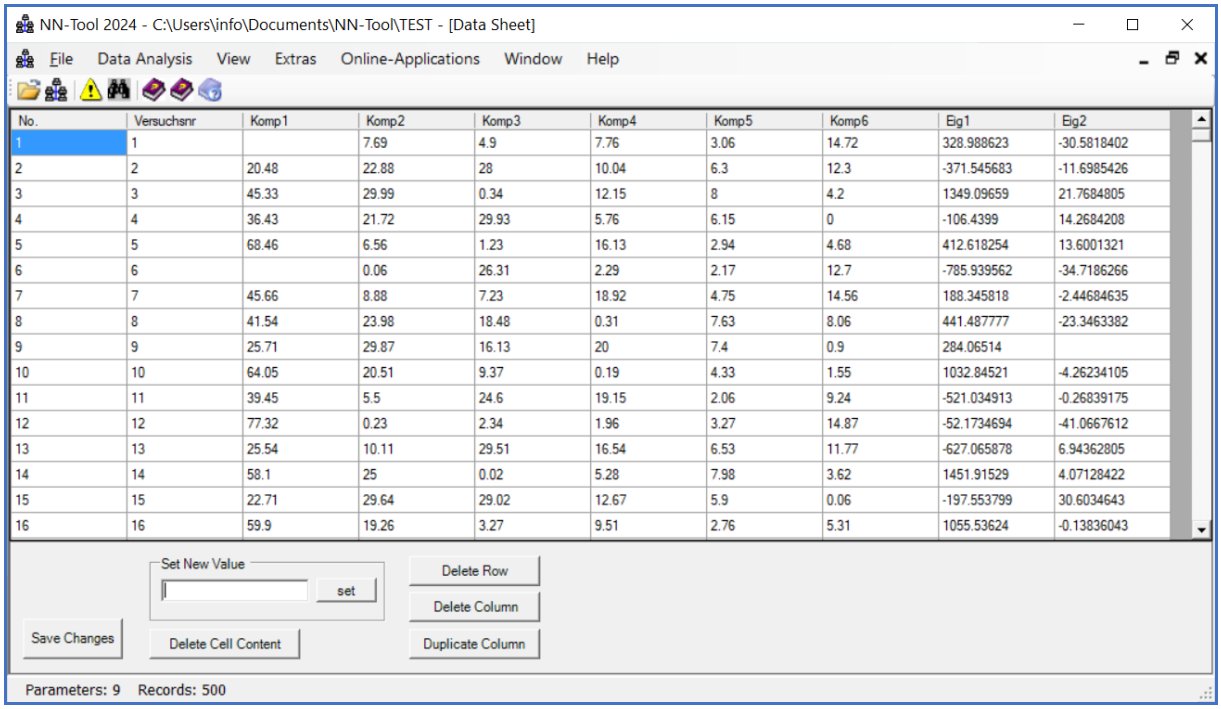
The NN-Tool data file can also be created directly (e.g. from databases), therefore, we are shortly going into the layout of the file in the following. The previously generated data file Test.pat has the following structure (equivalent to .csv):



In this example each line contains 9 parameters, which are separated by 8 semicolons. In the first line are **(always!**) the parameter names. For NN-Tool permissible parameter names are allowed to be of characters, numbers, parentheses ( ), the hyphen – and the underline _. Other type settings and special characters can lead to problems using the NN-Tool application modules. **In no case you should use brackets [], dot . and semicolon.** After the headline the first data row has to follow immediately. In the data lines 1 and 6 the second parameter does not have a value, in the 9th line the last parameter is missing. NN-Tool is able to process such incomplete records, as long as it is secured that in each line is the same quantity of semicolons. In case of maximum n parameters (n-1) semicolons have to be **in each line**. Additional blanks must not appear in numerical parameters (if necessary, remove them with

an editor), because they are interpreted as classifiers. In case of directly created input files please check your data by means of an editor before application of NN-Tool. **Invalid file formats are the most often application mistakes.**

As previously mentioned, NN-Tool creates the .pat-file from the Excel workbook, opens it, and determines initially the maximum number of occurring parameters and the number of available records. These values are displayed to the user in the status bar. In addition, the imported records are displayed in the window “Data Sheet”:



No.	Versuchsnr	Komp1	Komp2	Komp3	Komp4	Komp5	Komp6	Eig1	Eig2
1	1		7.69	4.9	7.76	3.06	14.72	328.988623	-30.5818402
2	2	20.48	22.88	28	10.04	6.3	12.3	-371.545683	-11.6985426
3	3	45.33	29.99	0.34	12.15	8	4.2	1349.09659	21.7684805
4	4	36.43	21.72	29.93	5.76	6.15	0	-106.4399	14.2684208
5	5	68.46	6.56	1.23	16.13	2.94	4.68	412.618254	13.6001321
6	6		0.06	26.31	2.29	2.17	12.7	-785.939562	-34.7186266
7	7	45.66	8.88	7.23	18.92	4.75	14.56	188.345818	-2.44684635
8	8	41.54	23.98	18.48	0.31	7.63	8.06	441.487777	-23.3463382
9	9	25.71	29.87	16.13	20	7.4	0.9	284.06514	
10	10	64.05	20.51	9.37	0.19	4.33	1.55	1032.84521	-4.26234105
11	11	39.45	5.5	24.6	19.15	2.06	9.24	-521.034913	-0.26839175
12	12	77.32	0.23	2.34	1.96	3.27	14.87	-52.1734694	-41.0667612
13	13	25.54	10.11	29.51	16.54	6.53	11.77	-627.065878	6.94362805
14	14	58.1	25	0.02	5.28	7.98	3.62	1451.91529	4.07128422
15	15	22.71	29.64	29.02	12.67	5.9	0.06	-197.553799	30.6034643
16	16	59.9	19.26	3.27	9.51	2.76	5.31	1055.53624	-0.13836043

In our example (application Test with data file Test.pat in the directory “C:\Users\info\Documents\NN-Tool”) a record consists of 9 parameters (even when they are not complete in every record). In total 500 records are available (the line with parameter names is not included). **This window should be used as a check for the correct data import!**

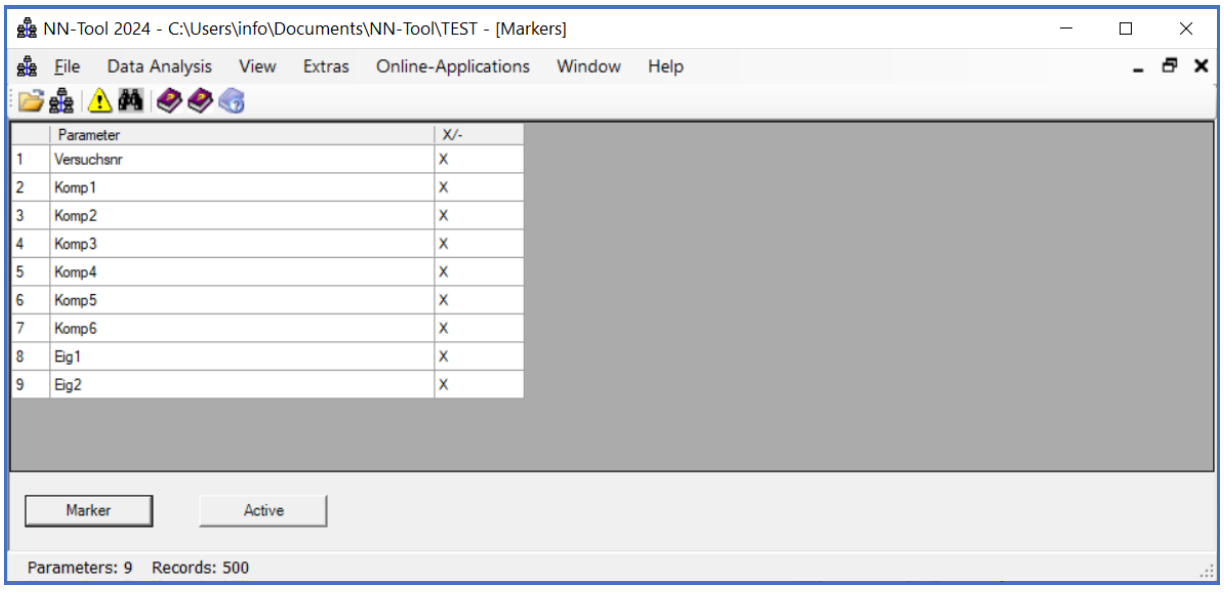
If the data import was not possible in the discussed manner, you have to use the other option:

2. Loading a .csv file

If a direct access from NN-Tool to Excel is not possible or the data are not available in Excel (but in a different application), you can import data as follows (here described for Excel again, but in other applications, the procedure is analogous):

- If both other options fail (i.e. the add-in cannot be installed or run, and a direct access to Excel is also not possible), it is necessary to convert the Excel data with Excel methods itself. To do so, convert the Excel spreadsheet to .csv-format by using **"Save As"** with the file type **".csv" ("character separated values“** or „comma separated values“). Then close Excel. If an indication of the nature of: *"File does not have the Microsoft Excel format. ...*", or similar appears, then click the corresponding button, i.e. **save the file in the .csv-format**.
- Now you have got a .csv file in which the data items are separated by a so-called list separator. This list separator depends (sorry!) on the Windows settings, specifically on the so-called Regional Settings. Using German regional settings, it is the semicolon, in English it is the comma. Now you can start importing the .csv-file in NN-Tool using **File/Load Datafile** (or press the appropriate button). In the "Open" dialogue set the file type to "CSV Character Separated Values (*.csv)". After that you have to specify the above-mentioned list separator. If your .csv file uses the semicolon (the default setting for German Windows versions), click the "OK" button. Otherwise, you must specify the comma (or possibly even another list separator). **Only one character** must be entered. Then NN-Tool imports the data file, creates an additional .pat-file, opens the window „Data Sheet“ and you can proceed as described in the following.

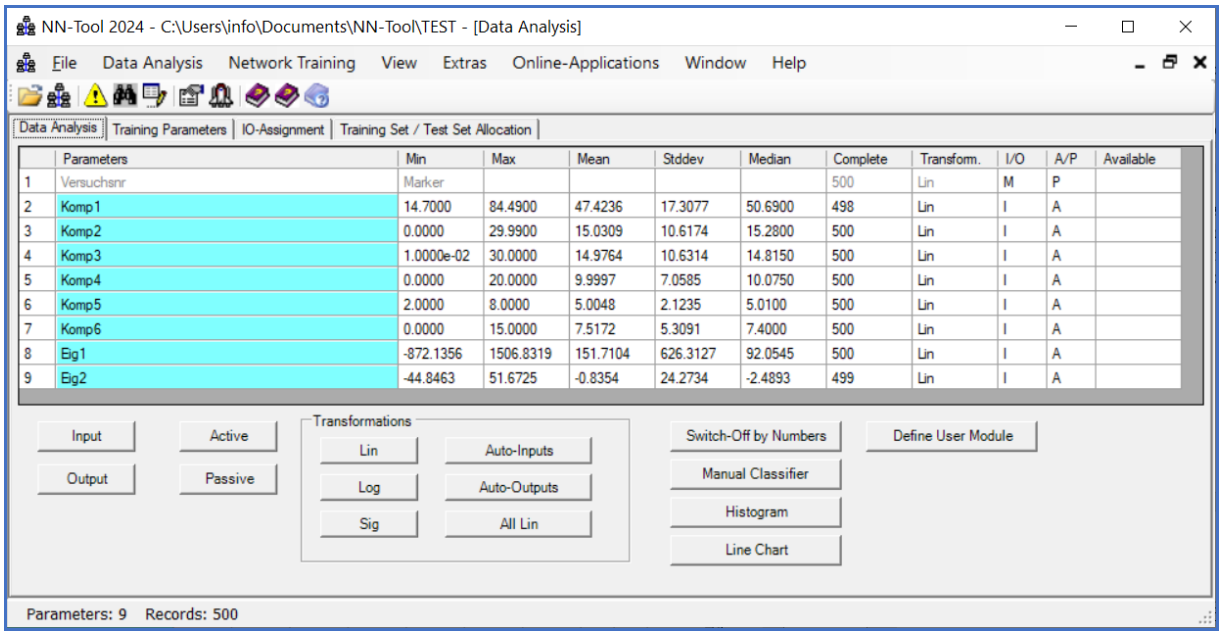
The next step involves working through the menu items in the **Data Analysis** menu step by step. The menu “**Data Sheet**” is used to control and possibly update the loaded records and has already been activated.
Close the “Data Sheet” window and run as a next step the menu point “**Deactivate Markers**” (or click the corresponding button):



Markers are known as parameters, which provide additional information about a data set, but are not to be included in the NN-model to be created. Test numbers, date and time etc. belong to this typically. These parameters should be disconnected immediately due to performance reasons. This is especially important when the values are not only numerical (e.g. date and time), because NN-Tool interprets these parameters as classifiers (the analysis of classifiers requires numerically a considerably greater effort than the analysis of continuous numerical

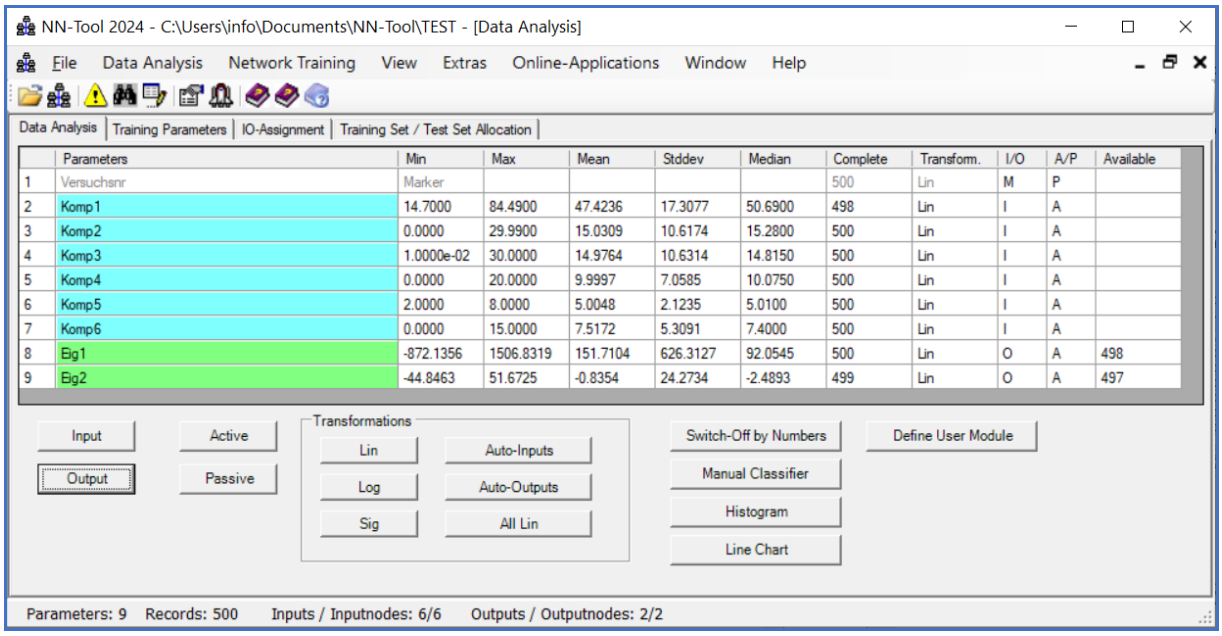
values). **Note: Shutdown of markers is not required. Parameters can also be turned off later. However, in case of large data sets and non-numerical values it is expedient.**
Click on the field Versuchsnr or the X behind it respectively and press the key "**Marker**". Then the parameter is noted as an inactive marker. With the key **"Active"** this setting can be revoked if necessary (this should not be carried out here).

Close the window and carry out the data analysis (menu **Data Analysis**, menu point **Analysis** or corresponding button respectively). With this step NN-Tool determines essential characteristic values of each parameter and displays these in the following window:



	Parameters	Min	Max	Mean	Stddev	Median	Complete	Transform.	I/O	A/P	Available
1	Versuchsnr	Marker					500	Lin	M	P	
2	Komp1	14.7000	84.4900	47.4236	17.3077	50.6900	498	Lin	I	A	
3	Komp2	0.0000	29.9900	15.0309	10.6174	15.2800	500	Lin	I	A	
4	Komp3	1.0000e-02	30.0000	14.9764	10.6314	14.8150	500	Lin	I	A	
5	Komp4	0.0000	20.0000	9.9997	7.0585	10.0750	500	Lin	I	A	
6	Komp5	2.0000	8.0000	5.0048	2.1235	5.0100	500	Lin	I	A	
7	Komp6	0.0000	15.0000	7.5172	5.3091	7.4000	500	Lin	I	A	
8	Eig1	-872.1356	1506.8319	151.7104	626.3127	92.0545	500	Lin	I	A	
9	Eig2	-44.8463	51.6725	-0.8354	24.2734	-2.4893	499	Lin	I	A	

After the first execution of the data analysis the window can be opened directly over the menu point "**Set Parameters"** (the analysis itself will not be executed again). For each parameter (which is not a marker) the minimum, the maximum value as well as the mean value, the standard deviation and the median (i.e. the value where 50 % of data are smaller and 50 % are larger) and the number of occurrence (point "complete") is displayed. By means of the data in the window you should firstly check whether the records have been read in correctly (Checking of plausibility). If parameters have been identified as classifiers you will find the key word Class in the field Min. This case is dealt with in detail in the next example.
The column Transformation indicates the used scaling and is made for the experienced user (see annex).
The column I/O indicates whether a parameter is input- or output value or marker respectively. Accordingly, the column A/P determines whether a parameter is active or passive. Only active parameters take part in the net modeling. A record only has to be complete regarding the active parameters. After activating of one or more parameters by mouse you can convert these settings with the corresponding keys. In our example the both last parameters are determined as output-parameters. Afterwards the window should look as follows:



	Parameters	Min	Max	Mean	Stddev	Median	Complete	Transform.	I/O	A/P	Available
1	Versuchsnr	Marker					500	Lin	M	P	
2	Komp1	14.7000	84.4900	47.4236	17.3077	50.6900	498	Lin	I	A	
3	Komp2	0.0000	29.9900	15.0309	10.6174	15.2800	500	Lin	I	A	
4	Komp3	1.0000e-02	30.0000	14.9764	10.6314	14.8150	500	Lin	I	A	
5	Komp4	0.0000	20.0000	9.9997	7.0585	10.0750	500	Lin	I	A	
6	Komp5	2.0000	8.0000	5.0048	2.1235	5.0100	500	Lin	I	A	
7	Komp6	0.0000	15.0000	7.5172	5.3091	7.4000	500	Lin	I	A	
8	Eig1	-872.1356	1506.8319	151.7104	626.3127	92.0545	500	Lin	O	A	498
9	Eig2	-44.8463	51.6725	-0.8354	24.2734	-2.4893	499	Lin	O	A	497

In the last column (**„Available**“) you will find the number of complete records available for each output parameter. A record is available for an output, when it is complete in all active input parameters and in the concerning output parameter. The record must especially not be complete in all outputs. For a good model, the number of available records should be as high as possible. In particular, the number should be significantly larger than the number of input parameters.
The status bar now displays additionally the number of active input and output parameters and the number of the corresponding network nodes. These figures differ only in the case of parameters of the so-called classifier type (see next example). As for a continuous (numerical) input or output parameter exactly one node is needed, both figures should be identical if only numerical (continuos) parameters are used (plausibility check). This is valid in our example.

Now close this window. With that this first simple data analysis is completed. We will come back to the large variety of further options later on.

After the data analysis you now have the possibility to create and train the neural network.

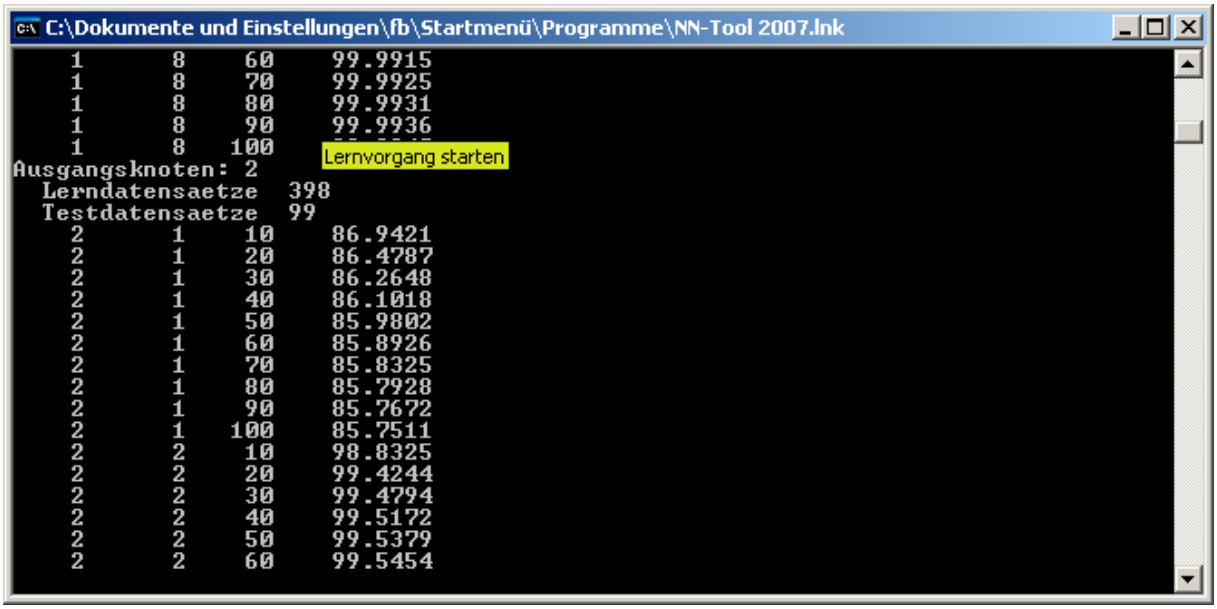
4. Network Training

Now start the learning process (the "training") of the network via the menu item "**Network Training/Start Training".** NN-Tool determines for each output parameter (here 2) automatically the best network structure and then binds the two separate networks together.

Allocation of records: For each output parameter the set of complete records is determined separately and is split in a training set and a test set. Using the default settings, each 5th complete record is assigned to the testset and the remaining 80% of complete records belongs to the training set. **Please note:** In the case of incomplete records, these assignments generally will vary for the various output parameters.

Training: NN-Tool will now determine for each output separately the optimal network structure. For this purpose, a number of different network structures (i.e., different number of internal nodes) and training steps („iterations") are tested for each output separately.

The number of internal nodes in the so-called hidden layer of a neural network essentially determines the modeling ability of the system. The more internal nodes are available, the more flexible is the network structure and the more complex relationships can be learned. With a large number of internal nodes increases, however, the danger of "overfitting", i.e. the network memorizes the data. In particular, the measurement errors will also be learnt.
So it is crucially to determine the right number of internal nodes. However, this figure is generally not known in advance. The correct number of internal nodes is determined by the complexity of the functions to be learned. If, for example, all the functions are linear in the input variables, one internal node per output would be sufficient. If the relationships are not linear, but still monotonous, you will need a few internal nodes only. Typically, such information is not available. In general, the complexity of realistic examples from process analysis or chemical research will be overestimated by the user. The functions often are monotonous, or at least monomodal (i.e. have only one minimum or maximum).
In a similar way the **number of training steps ("iterations")** affects the model accuracy. More steps usually improve the accuracy of the resulting model; but from a certain point (depends itself on the number of internal nodes) overfitting will occur.
Hence for each output parameter the optimum combination of internal nodes and iterations has to be found. **NN-Tool uses a grid search approach** for this purpose. The networks are trained on the records of the training set for various network structures and their modeling accuracy is determined on the independent test set. NN-Tool starts for the first output with a network with only one internal node and ten training steps, and then follows the first evaluation of this structure on the test set, then further evaluations after 20, 30, 40 up to 100 training steps. Then the same procedure for the first output parameter using a network with 2 internal nodes, then 4 and 8 internal nodes. The interim results will be displayed in the window:

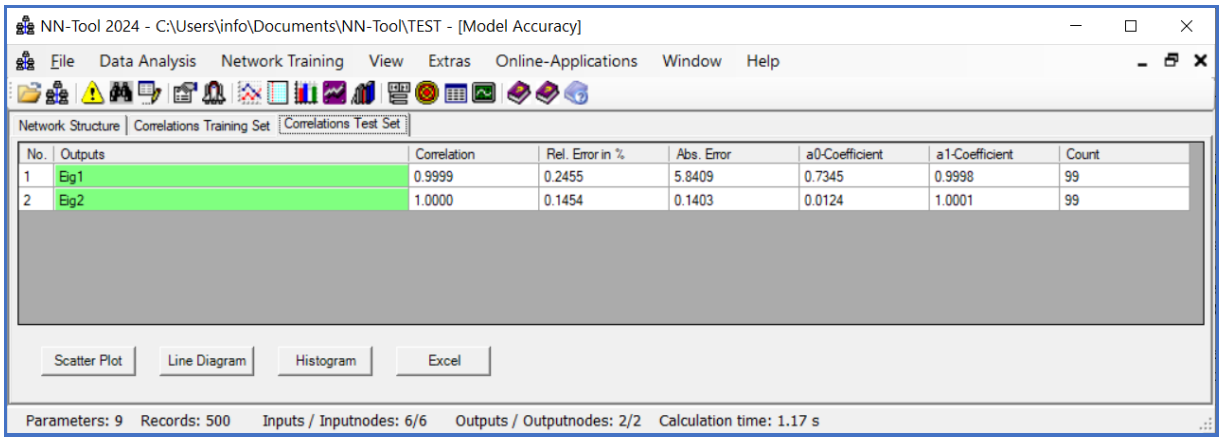


Here the training for the first output (column 1) has just terminated. For the first output the internal networks with 8 nodes (2nd column) and training steps from 60 to 100 ("iterations") are tested. This resulted in correlation coefficients of more than 99.99% (4th column). The term is explained in more detail later. A correlation coefficient of 100% represents a perfect model without errors, 0% means that nothing could be learned.

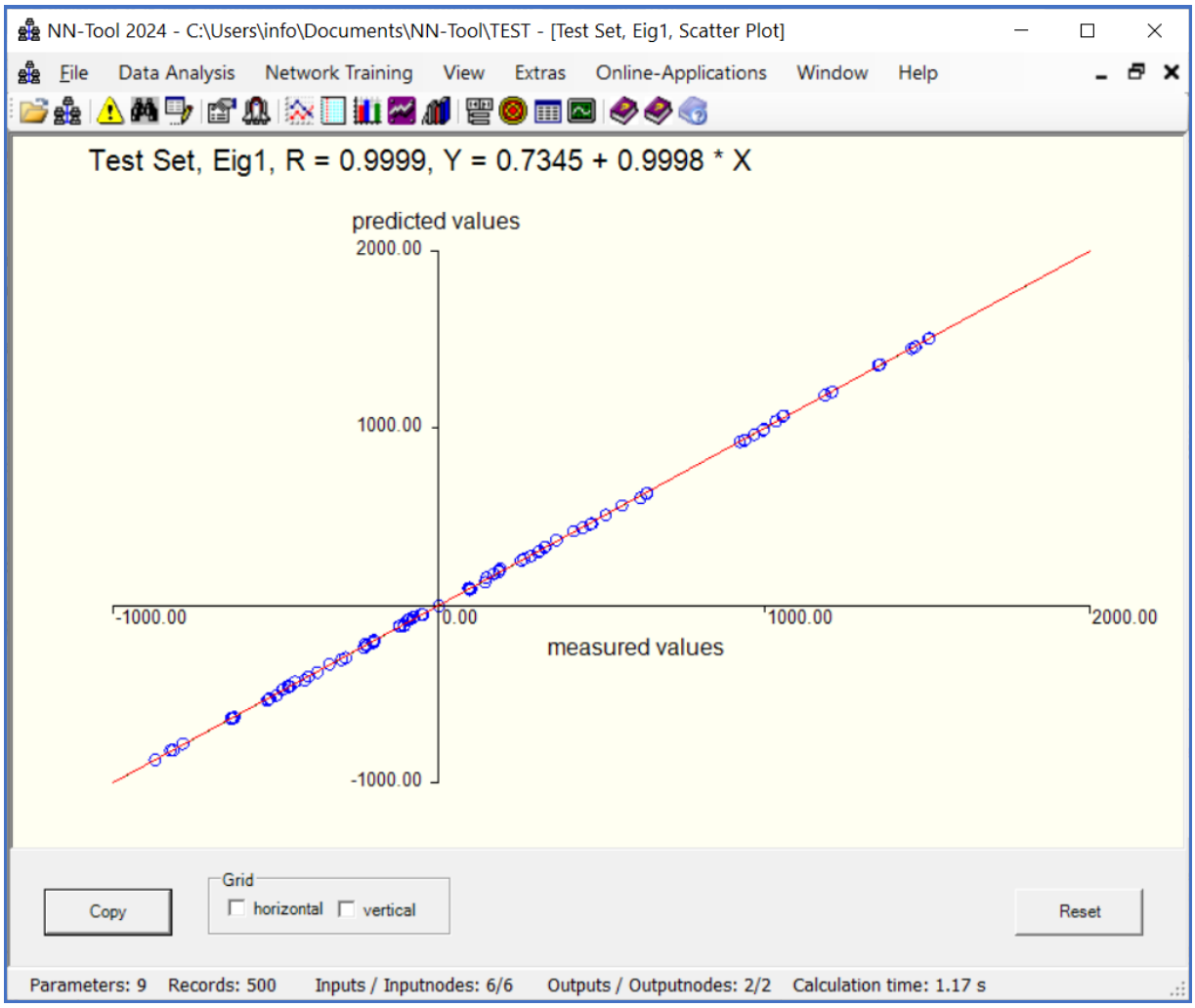
Following NN-Tool optimizes the network structure for the second output node (this corresponds to the output parameter 2, see also the following example). For this output we have 497 complete records of which 398 are used for training and 99 for testing.
The process starts again with a network with only one internal node and 10 training steps, etc.

If the entire process has finished, the optimum network structures for each output node (this corresponds to the output parameters in this example) are connected to one overall network.

Following NN-Tool executes a prediction for each single record of the training and the test set and then determines the **correlation coefficient** as well as the **relative absolute error** (in % of the scale variation) and the **absolute error** between the calculated and the measured values for each output parameter.
The coefficients of the linear regression line (**a0- and a1-coefficient**) and the number of records of the sets are displayed also. The meaning of the both coefficients is explained more detailed below. The correlation coefficients as well as the a1-coefficient are more often better measurements for the quality of a prediction than the relative error, because the distribution of data sets additionally flows in these coefficients. A perfect model corresponds to a correlation coefficient of 1, an a0-coefficient of 0 and an a1-coefficient of 1. The results are shown in the following window ("**Model Accuracy**"):



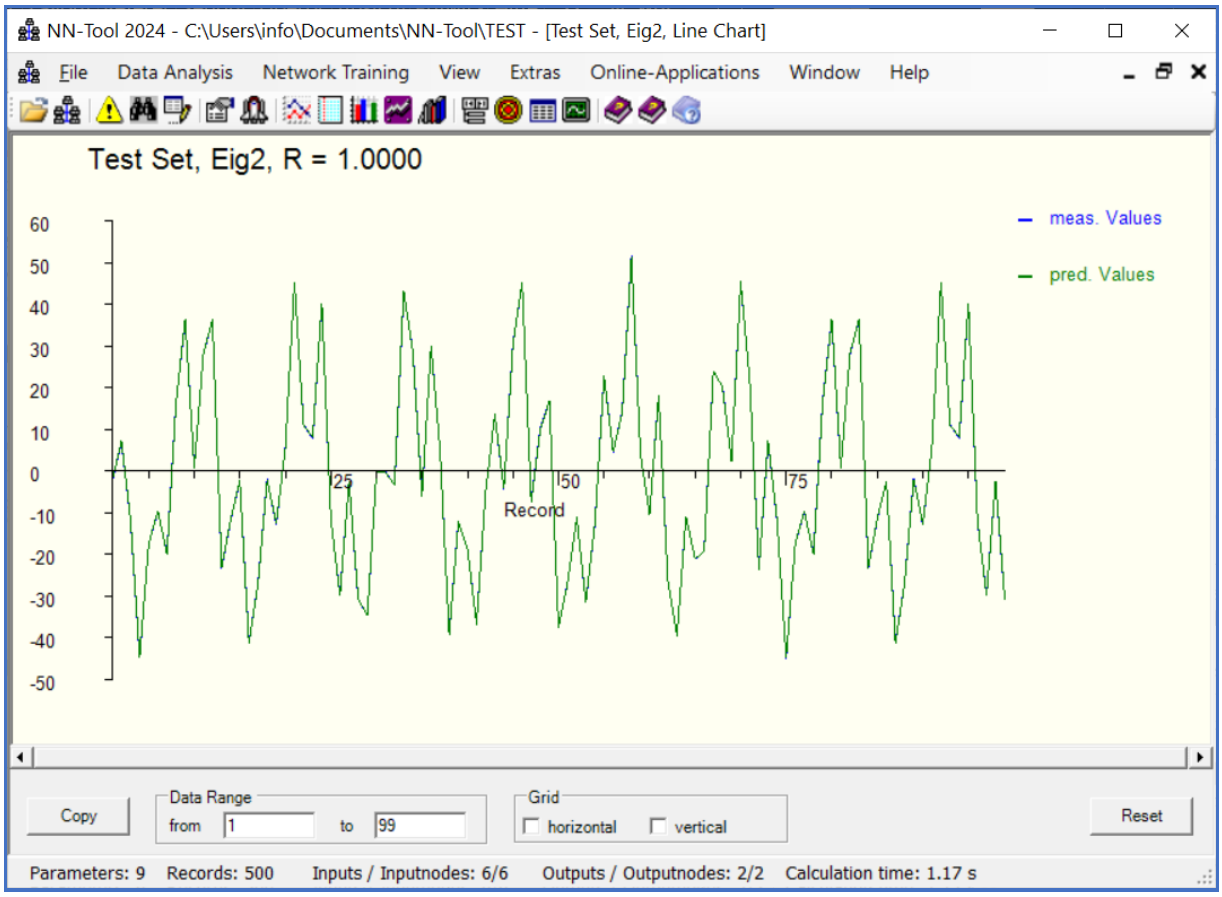
The colors of the parameters characterize the quality of modeling: "green" = good modeling, "yellow" = moderate, "red" = bad. The window can be displayed also manually at any time. For that one chooses in the menu **Network Training** the menu point **Prediction (Training and Testset)** (or push the corresponding button).
By clicking one of the output parameters can be selected and the following graphic (**Scatterplot**) can be displayed (also directly by using double-clicks):



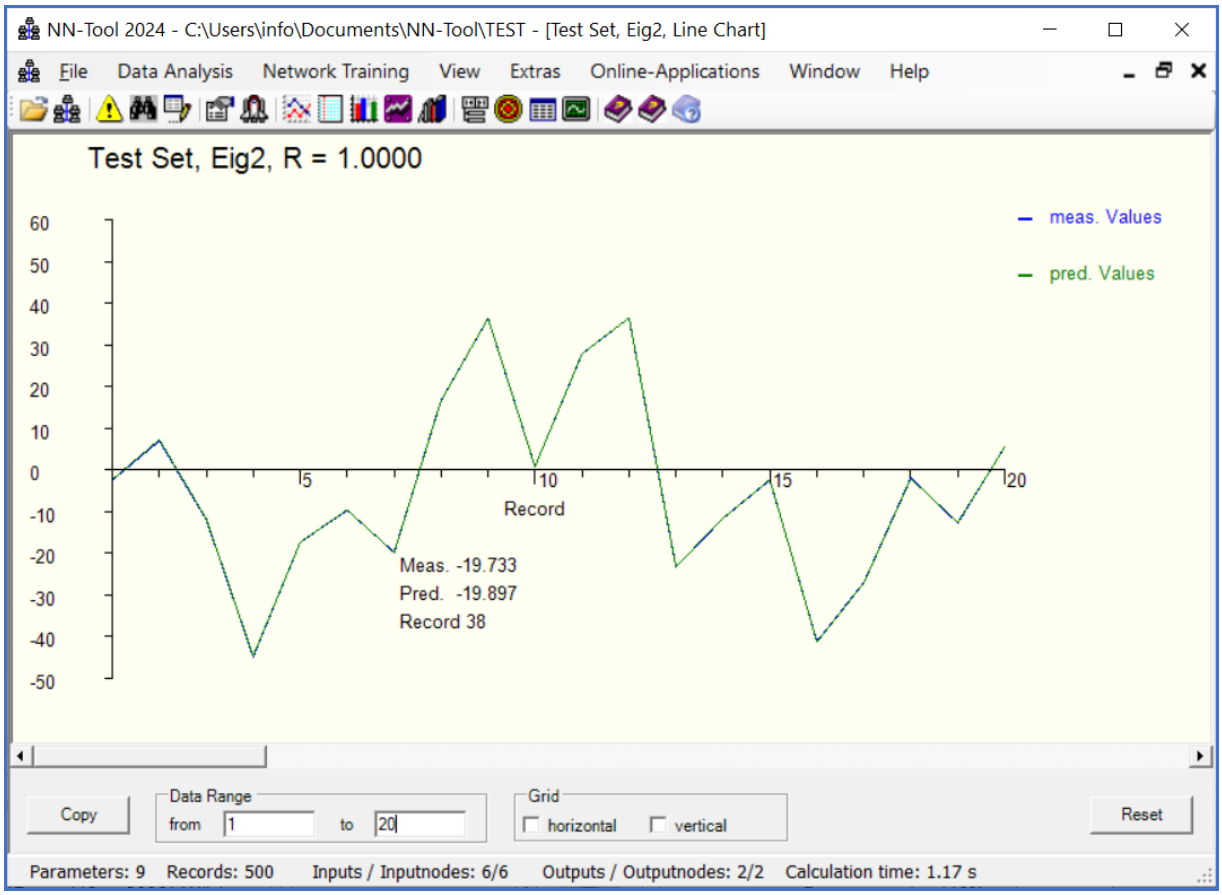
Here for each record of the test set the calculated (predicted) value is shown comparatively to the given (measured) value. Moreover the "45 degrees line" (blue line, here under the red line)

and the regression line (red line) is shown. a0-coefficient and a1-coefficient are defining the regression line. If the model is good the a1-coefficient has to be close to 1. Whereas the a0-coefficient has to be only small relating to the typical values of an output parameter. Therefore, a value of 0,73 is completely uncritical here. In the ideal case all points should be on the "45 degrees line" (i.e. on the straight line y = x) and the regression line should correspond to the "45 degrees line". In this way this representation makes possible a quick visually control of the net accuracy, which in case of more than 2 input values cannot be made easier comparable to any other graphical presentation. By clicking the key **Copy** the graphic can be saved to the clipboard and e.g. from there can be copied in a Word document.

Beside the Scatterplot also the **Line Diagram** is available for assessing the model accuracy (right mouse button):



In this chart the predicted and the measured values of a single record of the test set (or training set respectively) are displayed corresponding to the number of the record.
In both types of graphics additional information can be retrieved by clicking in the chart using the left mouse button. With the right mouse button, you can zoom in the graphic. Use the middle button to reset.
In the following the range of the first 20 test set records is displayed only. Additionally, one record has been selected (note: the seventh test set record is record number 38):



By pressing the key **Excel** in the window "Model Accuracy" the measured and calculated values of the output parameters are transferred to an Excel sheet. Thereby any formatted representations can be made.

These information can also be retrieved for the **training set** (window „**Model Accuracy**“, tab page „**Correlations Training Set**“). The tab page „**Network Structure**“ displays the selected optimal network structure for each output node.

If no sufficient model accuracy could be achieved, the learning process can be restarted using a large variety of parameter settings. This will be discussed in the next example. **It should be noted, however, that the principle attainable correlation coefficient is limited by measurement errors in the learning and test data**, because of error or correlation coefficient calculations always depend on comparisons with the existing data.
In this example, an almost perfect model could be achieved. This is of course a result of the specific sample data and not always true (see next example).

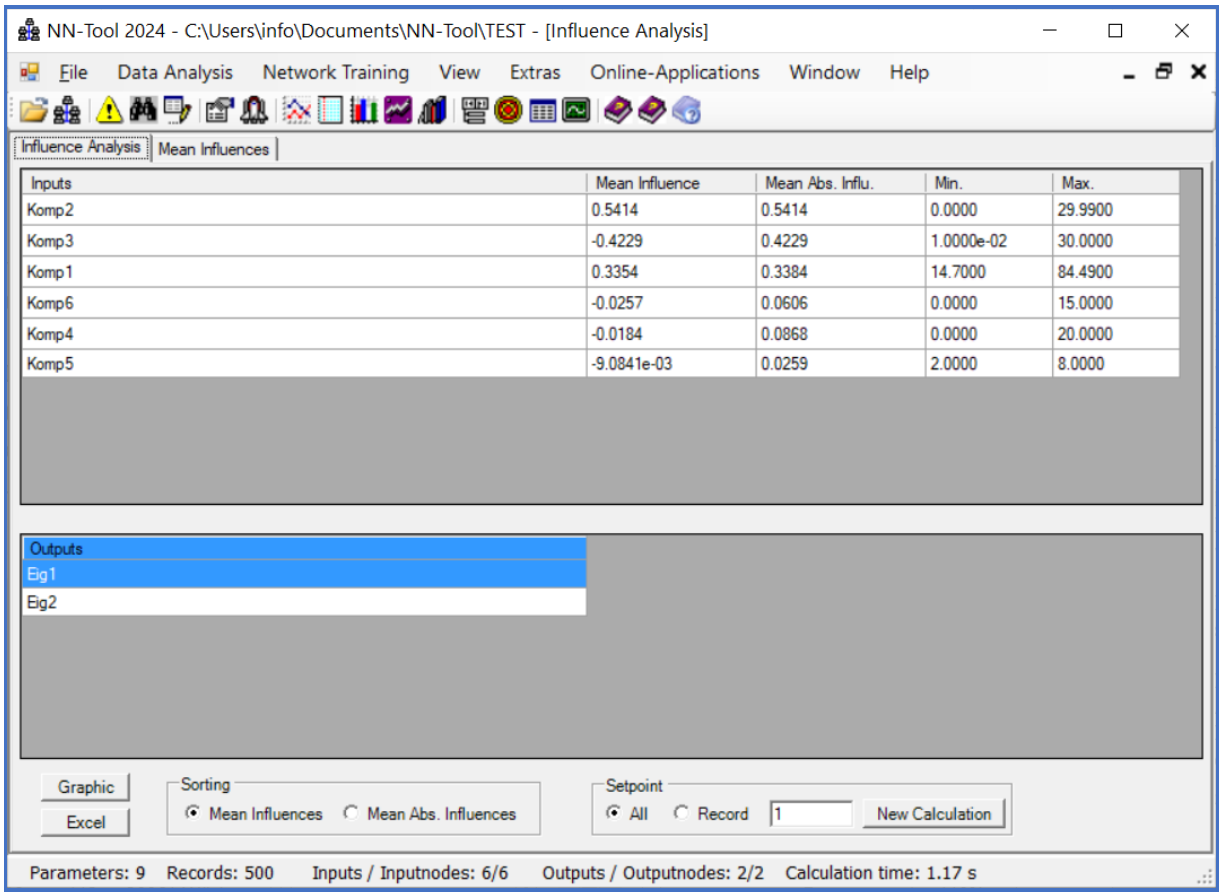
If a sufficient exact model is achieved, you can use this model now for analyses of the influence of the input values on the output values. Execute the menu point **“Influence Analysis”** in the menu **“Network Training”** (or push the corresponding button). The program now determines the mean influence and the mean absolute influence of an input value x_j on an output value y_i, by calculating and then averaging the corresponding partial derivatives of the NN-model (and the absolute value of that respectively) at all existing data points.

$$E_{i,j,mean} = \frac{1}{p}\sum_{i=1}^{p}\frac{\partial y_i}{\partial x_j}\ , E_{i,j,meanabs} = \frac{1}{p}\sum_{i=1}^{p}\left|\frac{\partial y_i}{\partial x_j}\right|$$

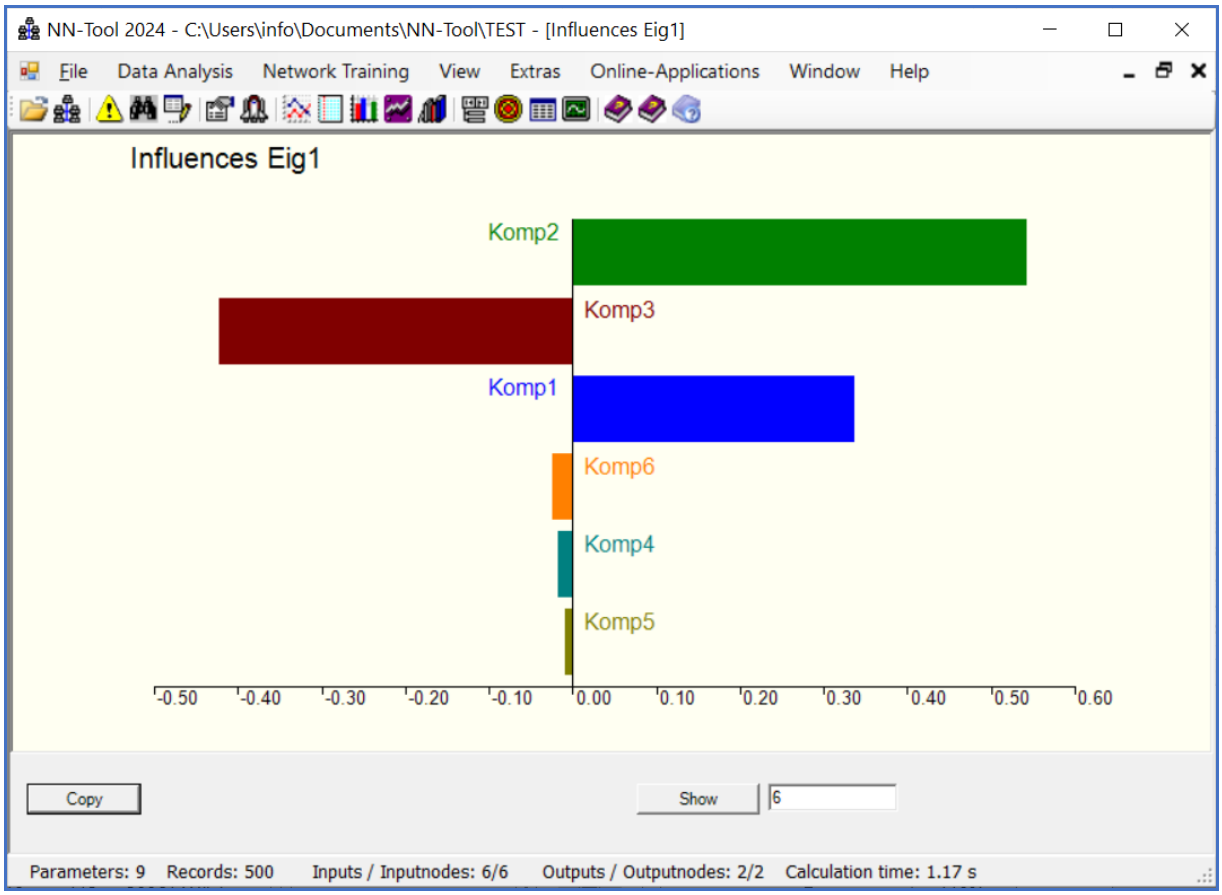
The mean influence is always equal or less than the mean absolute influence. In case of monotone dependencies applies:

$$\left|E_{i,j,mean}\right| = E_{i,j,meanabs}$$

However, if the value of the mean influence is really smaller, then the parameter x_j affects the parameter y_j one time positively and one time negatively (depends on the values of the other parameters).



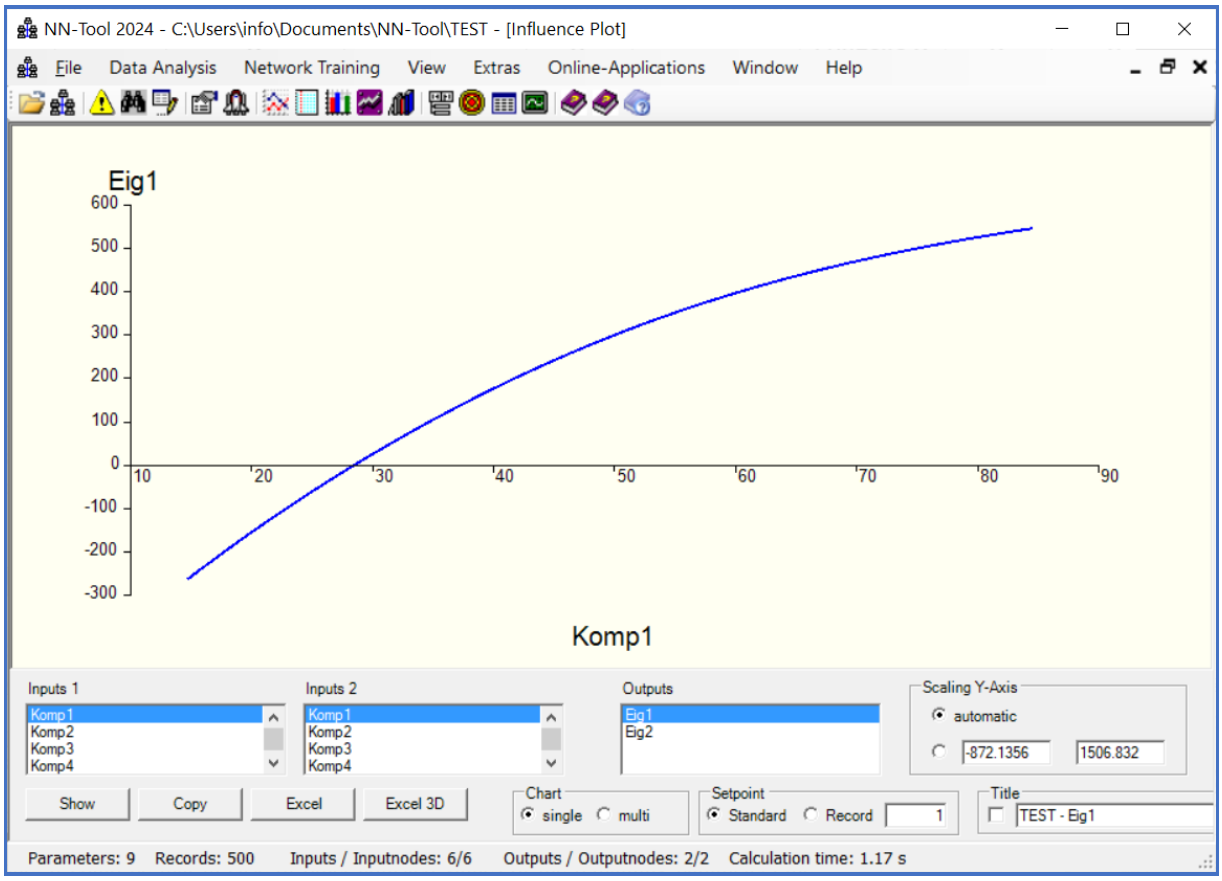
In the upper table the mean and mean absolute influence for all inputs are listed for the lower chosen output parameter. The indications min and max relate to the variation range of the parameters and provide only additional information. The example shows that the first two listed parameters "Komp2" and "Komp3" affect monotonically the output “Eig1”; all other parameters don’t have a monotone effect on the output "Eig1". The both most important influences are "Komp2" and "Komp3". An increase of "Komp2" has a positive effect and an increase of "Komp3" has a negative effect on the output "Eig1". By the key “**Graphic**” (or **double-click on the output name**) a corresponding graphical representation can be requested.



Using the option "**Setpoint**" the influences of the inputs on the outputs will be calculated at a fix determined record (the Setpoint), i.e. it will not be averaged over all data sets. For that firstly a record has to be selected and the button "**New Calculation**" has to be pushed. Using the option "**All**" and pressing "**New Calculation**" the average values will be calculated again.

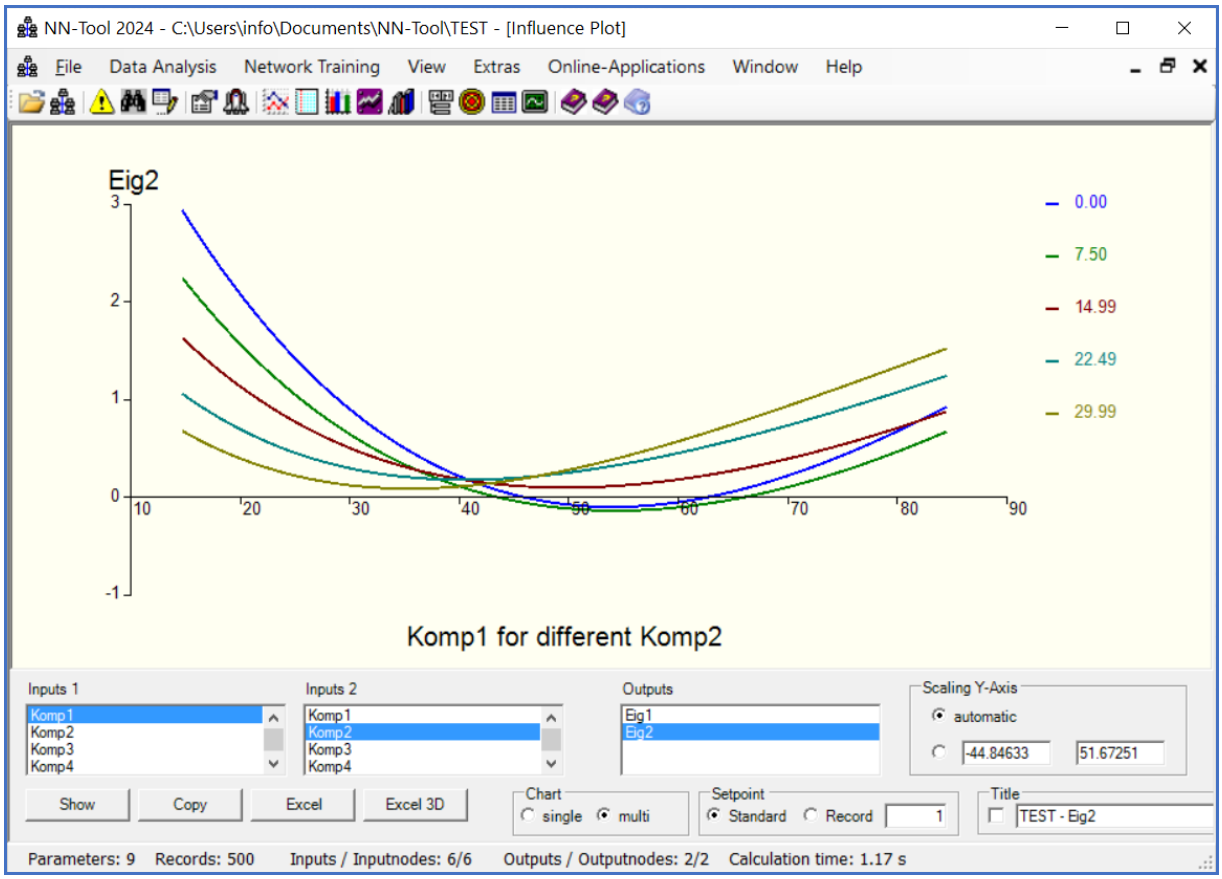
Important note / definitely try it out:
At this point you should make use of the "**Automatic Documentation**" option. This feature in the "**Network Training**" main menu makes it possible to generate all information relevant to a network fully automatically and save it in an **Excel folder with a variety of spreadsheets**. This includes graphics such as scatter plots and line diagrams.

By means of the option "**Influence Plot**" the influence of a single input value on an output value can be shown graphically:

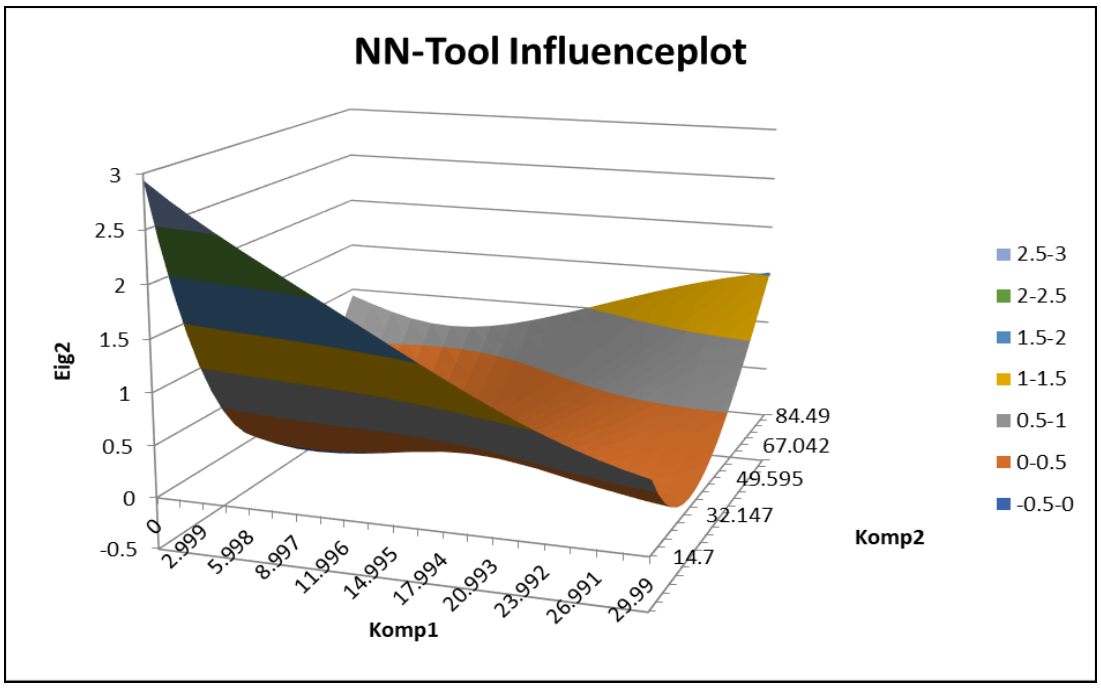


One input is varied by its scale range. At the same time all other inputs are kept on a setpoint (adjustable by the option Setpoint). As a standard the setpoint is determined by the mean values of all input values. As already known from the analysis before, with this adjustment of the setpoint "Komp3" has a negative effect on "Eig1".
Note: Only this chart alone is not able to show that the influence of "Komp3" on "Eig1" is monotone. It could be that in case of other adjusted values of the remaining parameters the influence of "Komp3" on "Eig1" would be positive.

By using the option multi or the selection of a second input parameter, the influence of two input parameters is displayed simultaneously:



With "Excel", the chart data will be transferred to Excel. With „**Excel 3D** ", a 3D- chart in Excel will be requested.

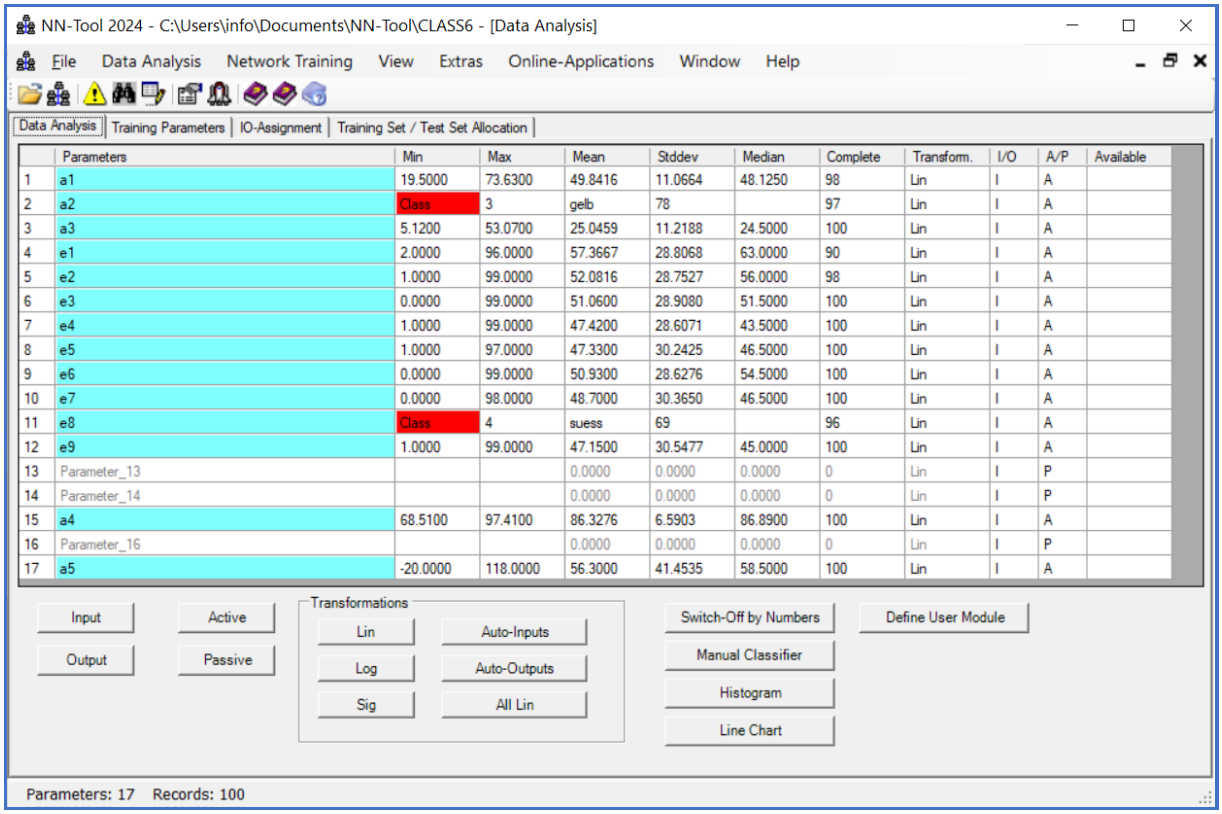


With the shown graphics firstly all input values have been set on their respective mean values. If other values are to be used, this is possible by the option “Setpoint”.

Reload network: Now exit NN-Tool. The created network can be loaded again at any time after a restart ("**File\Load Net**") or by double-clicking on the corresponding **".dsc" file** (description file).

5. Classifiers and Training Parameters

This section deals with further essential features of NN-Tool. For explanation another example problem is used with the application name **class6**. Re-start NN-Tool, generate from the Excel file class6.xls the corresponding data file class6.pat. Open and analyze the data file. Markers will not occur. The parameter description is as follows:



	Parameters	Min	Max	Mean	Stddev	Median	Complete	Transform.	I/O	A/P	Available
1	a1	19.5000	73.6300	49.8416	11.0664	48.1250	98	Lin	I	A	
2	a2	Class	3	gelb	78		97	Lin	I	A	
3	a3	5.1200	53.0700	25.0459	11.2188	24.5000	100	Lin	I	A	
4	e1	2.0000	96.0000	57.3667	28.8068	63.0000	90	Lin	I	A	
5	e2	1.0000	99.0000	52.0816	28.7527	56.0000	98	Lin	I	A	
6	e3	0.0000	99.0000	51.0600	28.9080	51.5000	100	Lin	I	A	
7	e4	1.0000	99.0000	47.4200	28.6071	43.5000	100	Lin	I	A	
8	e5	1.0000	97.0000	47.3300	30.2425	46.5000	100	Lin	I	A	
9	e6	0.0000	99.0000	50.9300	28.6276	54.5000	100	Lin	I	A	
10	e7	0.0000	98.0000	48.7000	30.3650	46.5000	100	Lin	I	A	
11	e8	Class	4	suess	69		96	Lin	I	A	
12	e9	1.0000	99.0000	47.1500	30.5477	45.0000	100	Lin	I	A	
13	Parameter_13			0.0000	0.0000	0.0000	0	Lin	I	P	
14	Parameter_14			0.0000	0.0000	0.0000	0	Lin	I	P	
15	a4	68.5100	97.4100	86.3276	6.5903	86.8900	100	Lin	I	A	
16	Parameter_16			0.0000	0.0000	0.0000	0	Lin	I	P	
17	a5	-20.0000	118.0000	56.3000	41.4535	58.5000	100	Lin	I	A	

This example contains two **classifiers** (tagged in color):

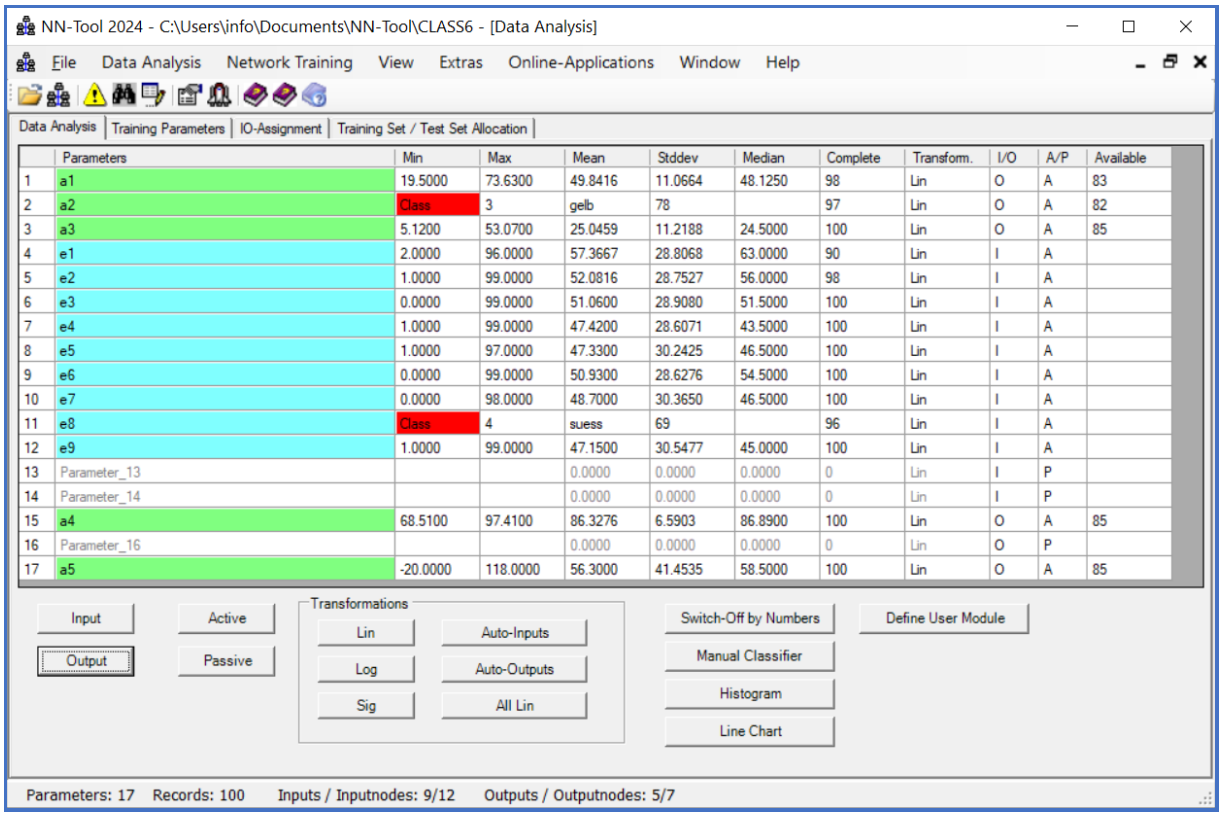
1. a2 with the classes (instances): „gelb“, „rot“ and „blau“ (German for "yellow", "red" and "blue")
2. e8 with the classes „suess“, „sauer“, „salzig“, „bitter“ ("sweet", "sour", "salty", "bitter")

NN-Tool identifies a classifier that way that in the corresponding data column are not only pure numerical values. On the other hand, every parameter which contains non-numerical elements (e.g. blanks) is treated as a classifier.
With the classifiers the following values are shown in the columns Min, Max, Mean, Stddev:

Column Min: keyword **Class**.
Column Max: Number of different classes (instances) of the classifier.
Column Mean: Most frequent class of the classifier.
Column Stddev: Number of arising of the most frequent class.

Furthermore, the example contains some completely empty columns called Parameter_13, Parameter_14, Parameter_16. These names have been given by NN-Tool. Parameters, which do not have at least two different values, are automatically deactivated by NN-Tool and also cannot be reactivated manually.

In this example the parameters a1-a3 and a4 and a5 are the output values. After switching over the result is:



	Parameters	Min	Max	Mean	Stddev	Median	Complete	Transform.	I/O	A/P	Available
1	a1	19.5000	73.6300	49.8416	11.0664	48.1250	98	Lin	O	A	83
2	a2	Class	3	gelb	78		97	Lin	O	A	82
3	a3	5.1200	53.0700	25.0459	11.2188	24.5000	100	Lin	O	A	85
4	e1	2.0000	96.0000	57.3667	28.8068	63.0000	90	Lin	I	A	
5	e2	1.0000	99.0000	52.0816	28.7527	56.0000	98	Lin	I	A	
6	e3	0.0000	99.0000	51.0600	28.9080	51.5000	100	Lin	I	A	
7	e4	1.0000	99.0000	47.4200	28.6071	43.5000	100	Lin	I	A	
8	e5	1.0000	97.0000	47.3300	30.2425	46.5000	100	Lin	I	A	
9	e6	0.0000	99.0000	50.9300	28.6276	54.5000	100	Lin	I	A	
10	e7	0.0000	98.0000	48.7000	30.3650	46.5000	100	Lin	I	A	
11	e8	Class	4	suess	69		96	Lin	I	A	
12	e9	1.0000	99.0000	47.1500	30.5477	45.0000	100	Lin	I	A	
13	Parameter_13			0.0000	0.0000	0.0000	0	Lin	I	P	
14	Parameter_14			0.0000	0.0000	0.0000	0	Lin	I	P	
15	a4	68.5100	97.4100	86.3276	6.5903	86.8900	100	Lin	O	A	85
16	Parameter_16			0.0000	0.0000	0.0000	0	Lin	O	P	
17	a5	-20.0000	118.0000	56.3000	41.4535	58.5000	100	Lin	O	A	85

There remain 9 active input parameters and 12 input nodes are allocated to them, because the classifier "e8" needs for its 4 classes 4 input nodes for coding.
Corresponding for the 5 active output values result in 7 output nodes.
Although the additional functions for the training parameters are not required for this application, they are introduced here shortly:

Transformations / Histogram: these buttons allow for non-uniformly distributed parameters a transformation to come to better distributed parameters (see Annex: Non-linear Scaling Transformations). These transformations can be set individually for each parameter using the buttons **Lin, Log** and **Sig**. With the key **histogram** (or double-click on a parameter) a graphical representation of the distribution of the selected parameters is displayed. In this chart the effect of transformations can also be studied (try!). Using the keys **Auto-Inputs**, **Auto-Outputs**, NN-Tool determines the best scaling-transformation to choose for each parameter (separately for inputs and outputs). Using the button "**All Lin**", all parameters will be reset to the linear scaling transformation (default case).

Switch-Off by Numbers: In case of datasets with a lot of parameters this function allows for setting all parameters passive with just one button, which have not been measured sufficiently often.

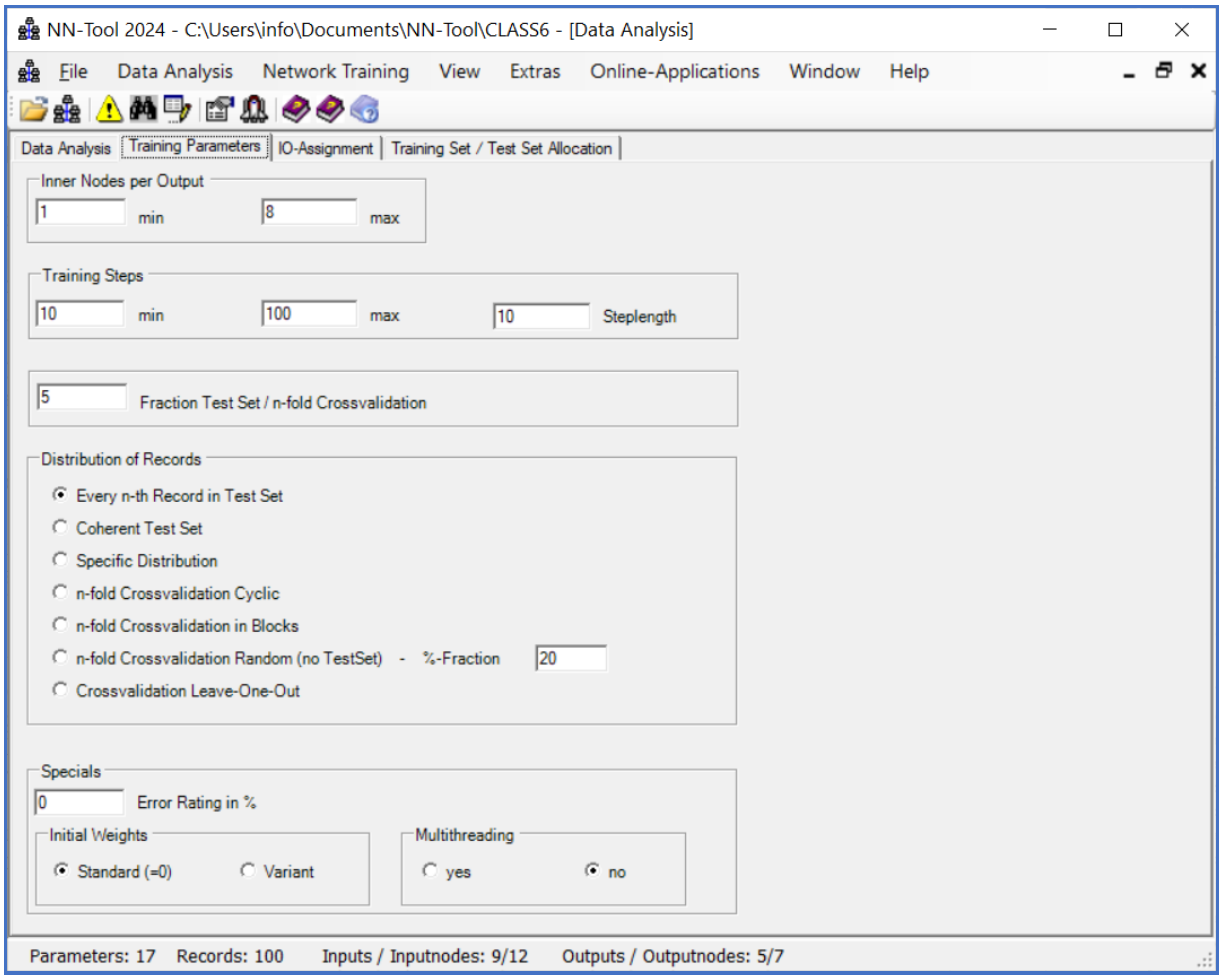
Manual Classifier: As already mentioned, NN-Tool identifies a classifier by arising of non-numerical parameters. However, it can occur that the instances of a classifier are numerical values themselves, e.g. classifier catalyst-type with the values 1, 2, 3, 4. Such a classifier must be changed manually. For that the corresponding parameter is tagged and then the menu point

is executed. This can be reversed by executing the analysis again.
Note: If it is intended to use the neural net as an Excel-Add-In, this option should not be used! In contrast to NN-Tool, Excel does not distinguish between a number and a text, which looks like a number. In such cases these parameter values should be turned into classes by attaching a character before or behind the value.

Line Chart: By selecting a parameter and pressing the button „**Line Chart**“ (or right mouse click) a graphical representation of the values of the selected parameter is displayed (try it!).

Tab Page Training Parameters

In Example1 (application „Test“) for each output parameter networks with 1, 2, 4, 8 inner nodes and 10, 20, 30 ... 100 learning steps have been tested.
This tab page allows a more precise setting of the learning process:



The automatic training of NN-Tool creates an optimum net for each output node, i.e. for each function to be learned. The created nets will be finally joined to an overall net. This leads to a better **prediction quality** of the nets, as if all output values are optimized together. With that for each output parameter the number of complete records is determined also. That means that a record which is incomplete in one or more output values, will still be used for net training for all other output values (however, the input values have to be complete, because of the uniqueness of the problem). These methods guarantee that the maximum of information is got

from the data set. The user only gives a **frame for the optimization**. Within this frame the best net each per output node is found out. The net quality is determined by the error of predictions on the testset. With the tab page **"Training Parameters"** you specify the above-mentioned frame. In detail the fields show:

- **Inner Nodes per Output**: These values determine the minimum and maximum size of subnetworks, e.g. 1 and 32. In this case the system checks the following net sizes for each output parameter (or nodes with classifiers respectively): 1, 2, 4, 8, 16, 32, i.e. the net size is doubled each. (An alteration from 10 to 11 inner nodes does not bring a considerable change of the training ability and due to this they are doubled each.) Therefore, the total network can consist of 7*32 inner nodes (a maximum of 32 inner nodes for each of the 7 output nodes). But usually, the total networks are considerably smaller.

- **Training Steps**: These values give the lower and upper limit of the number of training steps and the number of steps, according to which the test set is being evaluated. The setting 10, 100, 10 means: The system executes at first 10 training steps (minimum number), following the quality of the net is tested, then 10 further training steps are done (Steplength) and tested again etc. until the maximum number of steps has been executed. This will be repeated for each output node and each setting of inner nodes.

- **Fraction Testset / n-fold Crossvalidation**: Indicates the fraction of complete records of an output parameter which is to be used for testing (i.e. not for training). A 5 means that every 5^{th} complete record is used for testing. **Please observe:** this can be another allocation for each output parameter, if the output values are incomplete. If training is performed in crossvalidation mode (see below), the value indicates the number of different test sets.

Distribution of Records: In some applications, the quality of the resulting network models depends sensitively on which records have been used for training (training set) and which have been used for testing. This is particularly the case when only a few records are available or the records show a high correlation to each other (e.g., for time series applications). For this reason, the following options for the partitioning are available:

- **Options: every n^{th} Record in Test set / Coherent Test Set:** indicates how the datasets of the testset are to be chosen from the available records for an output node. For test series the option "Every n^{th} Record in Test Set" is more useful, for time series the option "Coherent Test Set" should be preferred, because of the typically high correlation of neighboring records. NN-Tool chooses the corresponding option automatically, i.e. also here normally no change is necessary.
- **Specific Distribution:** Allows detailed assignment of every record to the test set or the training set respectively by using the tab page **"Training Set / Test Set Allocation"**. Furthermore, it is possible to assign a weight to every record. This weight indicates how often a record will be used (0 = record ignored).
- **Crossvalidation:** Allows using of multiple, dynamically created test sets for every output. This option reduces the dependence of the resulting network structure from the selection of a specific test set. If for example the value of the field n-fold Crossvalidation is 5 then 5 partitions of learning and test set are made for each output

parameter. For each of these partitions, the entire test program in terms of internal nodes and the training steps is executed automatically. Then, each output node will be trained with the optimal network structure on all records. Finally, the individual sub-networks will be joined to the overall network. **Note:** A specific weighting of records in tab page "Training Set / Test Set Allocation" will also be taken into account using crossvalidation. For the dynamic allocation of the records **four different strategies** are available:

- **n-fold Crossvalidation Cyclic**: With this option and n = 5, the first dynamic test set contains the following complete records: 1, 6, 11, .. the second test set consists of records 2, 7, 12, .. etc. This partitioning is generally especially suitable for uncorrelated data series.

- **n-fold Crossvalidation in Blocks**: In this option, and n = 5, the first dynamic test set contains the first fifth of the records; the second test set contains the second fifth, etc. This partitioning is generally particularly suited for **time-dependent data** (time series, see corresponding example). If it is unclear whether neighboring data sets are correlated, this setting is recommended, i.e. **if in doubt you are on the right side with "Crossvalidation in Blocks".**

- **n-fold Crossvalidation Random**: In this option, and n = 5, each of the five test sets consists of records which are assigned to the test set with a probability specified in the field %-Fraction. In this mode, no final assessment of the test set accuracy can be carried out (only training set).

- **Crossvalidation Leave-One-Out**: The extreme case "Leave-One-Out" creates as many different test sets as there are complete records for the output under consideration. In this case every test set consists of only one record (no correlation information is possible during training runs). This mode is by far the most expensive in terms of calculation time, but can - at least for uncorrelated data – achieve the highest model accuracy. Especially for small data sets of (hopefully) uncorrelated test series suitable. In time series, this option - because of the generally high correlation of records among each other - is not recommended.

- **Error Rating in %:** As already mentioned NN-Tool tests for each output parameter singly the optimum net configuration with different numbers of training steps and inner nodes. Finally the net is used which achieves the best prediction precision on the test set. Normally for that the mean relative errors of the single nets are compared. With this option punishment terms can be additionally put on networks having a lot of inner nodes. A value of x > 0 with this option means that the mean relative error of a fraction net with n nodes is multiplied by a factor of 1+ (nx / 100). This will be used for the network choice. This option is of importance, when a high correlation of testing records with corresponding training records has to be assumed. In this case the test set error itself does not offer a sufficient basis for assessment.

- **Initial Weights:** The learning algorithm of NN-Tool is like all other similar methods an iterative process in which on the basis of initial values for the weights of the network gradually always better solutions are to be found. Such a procedure requires an

initialization of the network weights. For NN-Tool all weights are initially set to zero. **Note:** in the literature you will often find the statement that this setting is always a local minimum of the learning process, and that learning would not converge. For the NN-Tool learning algorithm this is generally not the case. The author knows only one super-symmetrical example (i.e. XOR) to which this standard initialization is indeed a local minimum and NN-Tool does not converge. In such cases, choose „Variant“ for „Initial Weights“ and thus convergence can be achieved also.

- **Multithreading:** This option enables the learning algorithm of NN-Tool to use several processing cores (if possible) to achieve a higher calculation speed. **The option is useful only under the following circumstances:**
    1. The PC possesses two or more cores.
    2. There are more than one outputs to train or option Crossvalidation is used
    3. Training is time consuming because of a large number of records or parameters.

Tab Page IO-Assignment

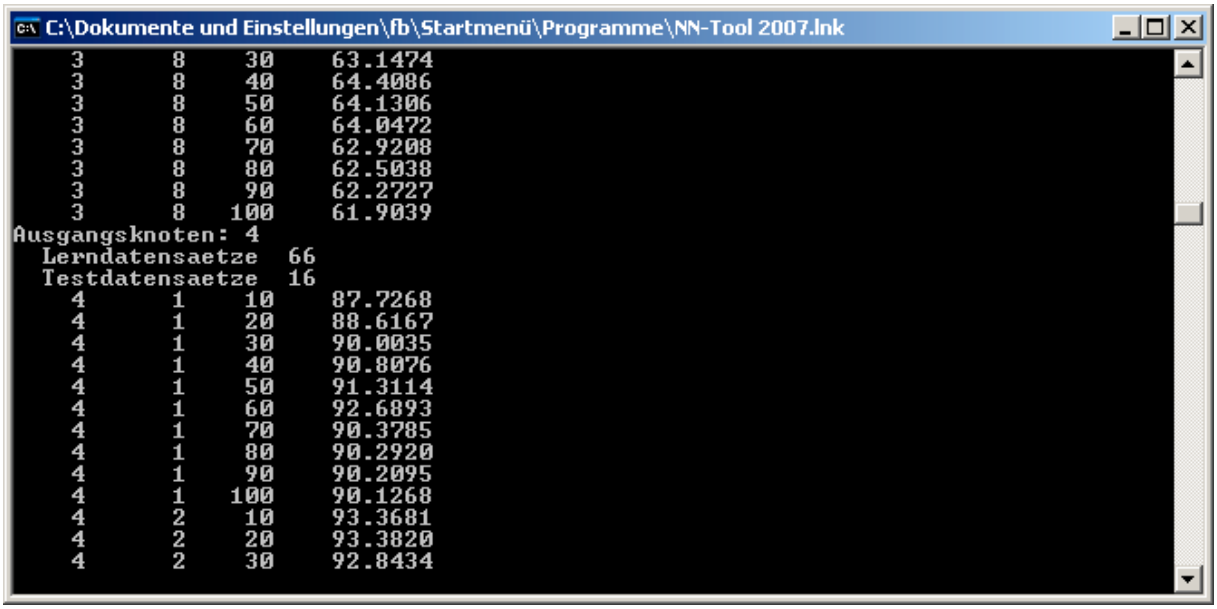
Allows allocation of input- to output parameters. As standard each active input parameter is enlisted for modeling each output parameter. By means of this function NN-Tool can be instructed to leave certain inputs out of account with certain outputs or to allow only a monotonous relation between a certain input and a certain output. This affects correspondingly the number of complete records of an output parameter. I.e. a seldom measured input parameter, which is deactivated for a certain output, does not reduce the number of complete records which are available for this output. The function IO-Assignment affects only the automatic training procedure.

Tab Page Training Set / Test Set Allocation

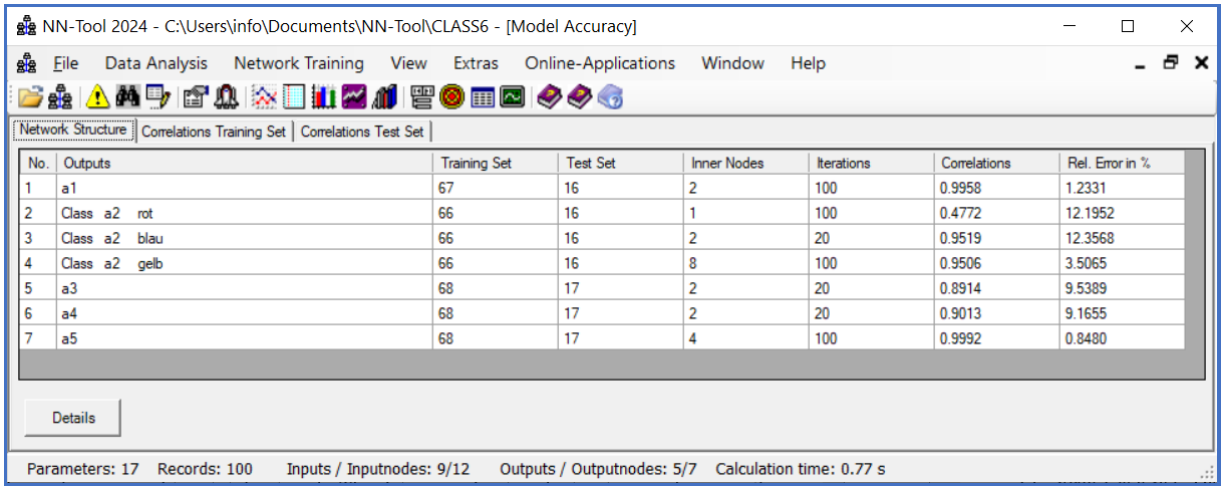
Allows you to assign individually each record to the training set or the test set. Of course, this assignment is valid only, if the option „Specific Distribution“ has been selected at "Distribution of Records" on the tab „Training Parameters“. In addition, a weighting of the records can be made.

These functions are not needed here, but maybe you experiment with that. The changes can be cancelled by executing the analysis again.

We adopt the **default setting** and start the automatic training.



A window is opened as shown above (If it happens too quickly: this information can also be accessed afterwards, see below). In this display the optimization of the 3rd output node has been just finalized and the 4th optimization has started (1st column). Regarding the 4th output node 82 records are complete and 66 of these are used for training and 16 for testing the fraction network for the 4th node. At first the network is tested with one inner node (2nd column) after 10, 20, 30... training steps (3rd column). The corresponding correlations (in %) for the prediction on the testset are listed in the 4th column. After running through the best networks each are joined to a total network. The determined network structure is shown in the following window (**Note: By default, the tab page "Correlations Test Set" is activated**):



Furthermore the whole information is written in a file called class6.pro. **The file can be displayed directly using the "Details" button.** For each output node you can find the description for the best single network each in the window (or the file respectively). For instance, for the third output node (corresponds to class "blau" ("blue") of classifier "a2") results an optimum network with 2 inner nodes with 20 training steps. This network has a correlation of 95.2 % and a relative error of 12.36% on the testset. 66 records have been used for creating the corresponding fraction network. The error calculations are based on the 16

records of the test set. Then the seven single networks have been joined to one total network with 21 inner nodes.

You can get a complete documentation of your modeling in the main menu “Network Training”, menu item “Automatic Documentation”. Definitely try it!!

The network can then be used to create influence plots or for optimization purposes using the integrated application modules (see chapter "Application Module Control Panel" and chapter "Complex Optimization Problems").

In the next chapter we will look at **options for improving the model accuracy**.

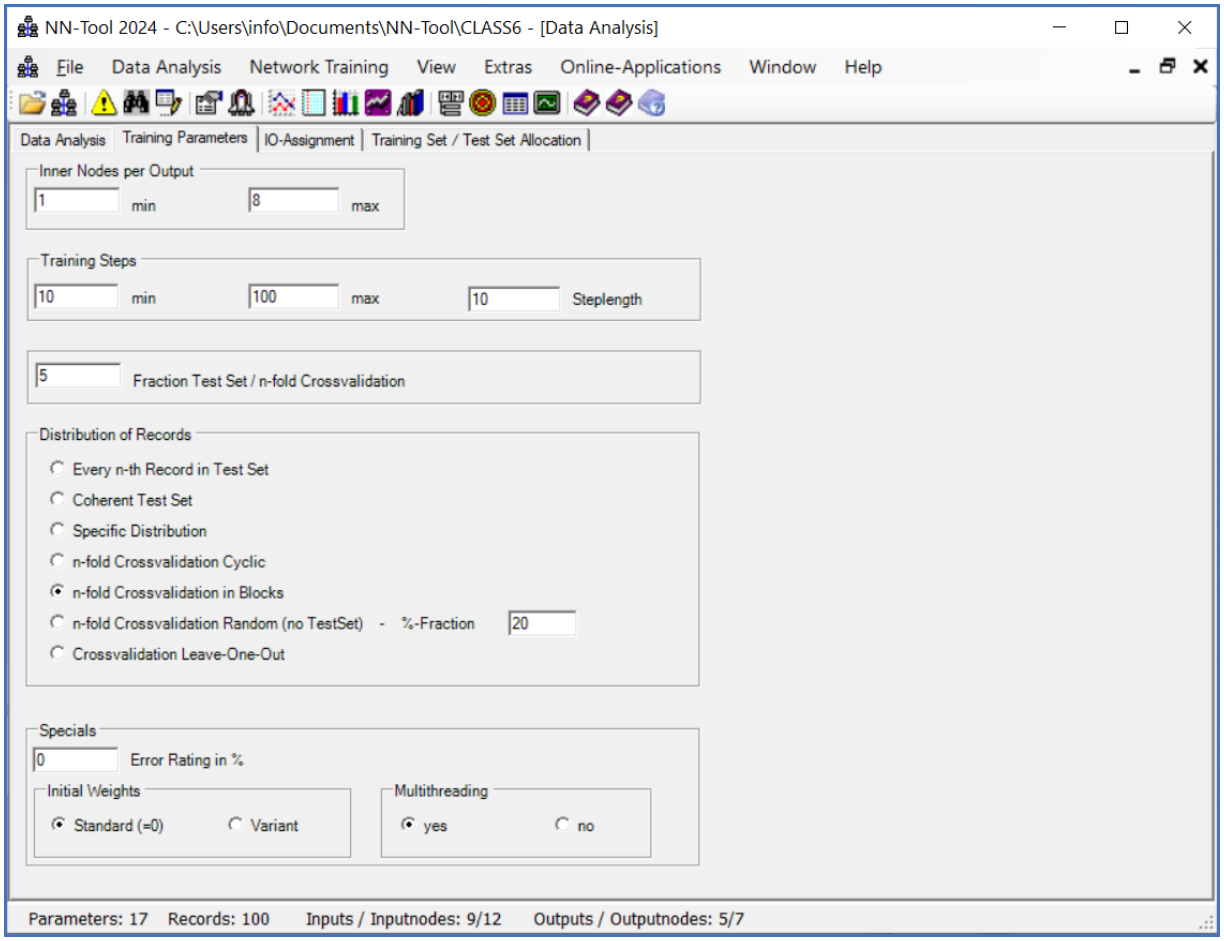
6. Optimization of Model Quality – Crossvalidation and Input Optimization

In the previous example "**class6**" we had the unfavorable situation that there were relatively many inputs (input parameters) and only relatively few data sets.

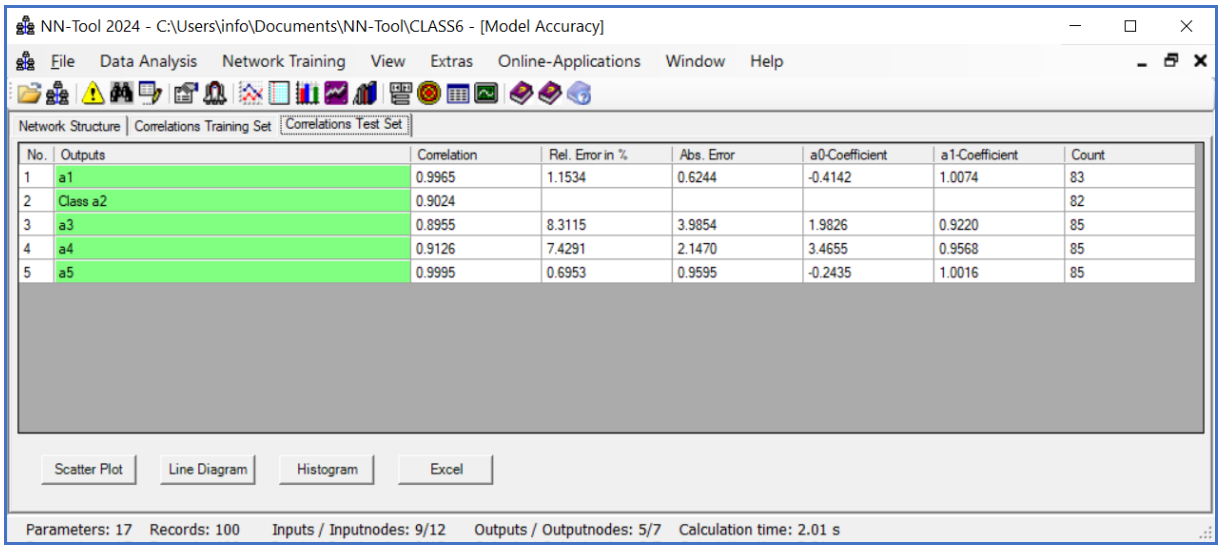
In principle, for good modeling, the ratio of the number of records to the number of inputs should be as high as possible. You can never actually have too many datasets (records).

A small number of data sets also means that the **errors and correlation coefficients determined are statistically less certain**. We will now tackle the last point first by moving on to Crossvalidation.

First load the "**class6**" network created previously (Chapter 5) and switch to "Network training \ Set Training Parameters". Select the "**Crossvalidation n-fold blockwise**" option (we are not sure whether neighboring data sets are correlated) and the "**Multithreading**" option:



Now start the training process:

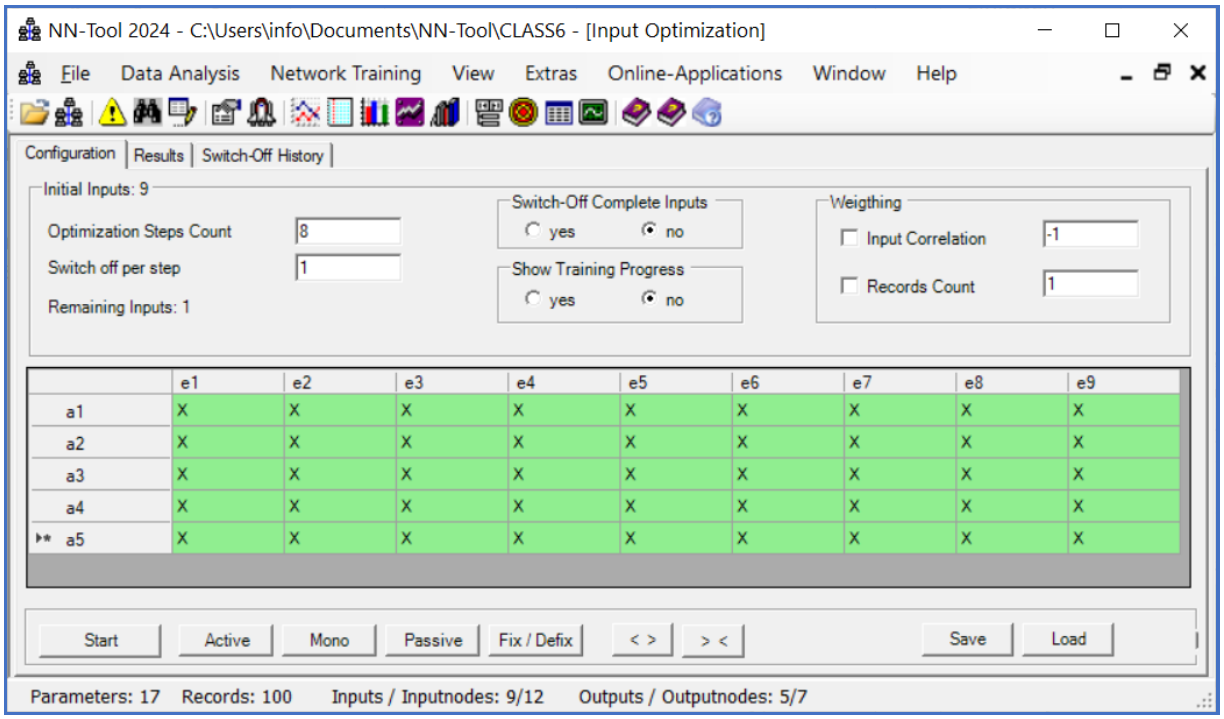


We initially achieved a (slight) improvement in the model quality. Above all, the errors and correlations determined on the test set are now better statistically verified ("Count" field).

In the next step, we want to improve the **ratio of the number of data sets to the inputs**. Since we cannot carve out data sets, we can instead try to reduce the number of inputs. This is where the "**Input Optimization**" option comes into play.

This option in the Network Training main menu makes it possible to successively (sequentially) reduce the number of input parameters for the various outputs in order to find an **optimal input parameter configuration**. The goal is to only keep the necessary inputs for each output and to switch the remaining ones passively (see IO-Assignment). In contrast to the more common "**Optimum Inputs**" methodology (Appendix 4), which is based on preliminary correlation analyses, here the optimal inputs are determined on the basis of networks that have already been created. In contrast to the "Optimum Inputs" methodology, **non-linear influences of the input parameters** are also taken into account. The method is therefore much more general (and generally more accurate) than the usual approach of parameter selection based on correlation coefficients (see "Optimum Inputs"). Nothing is free. It is also significantly more computationally expensive. However, handling is extremely comfortable.

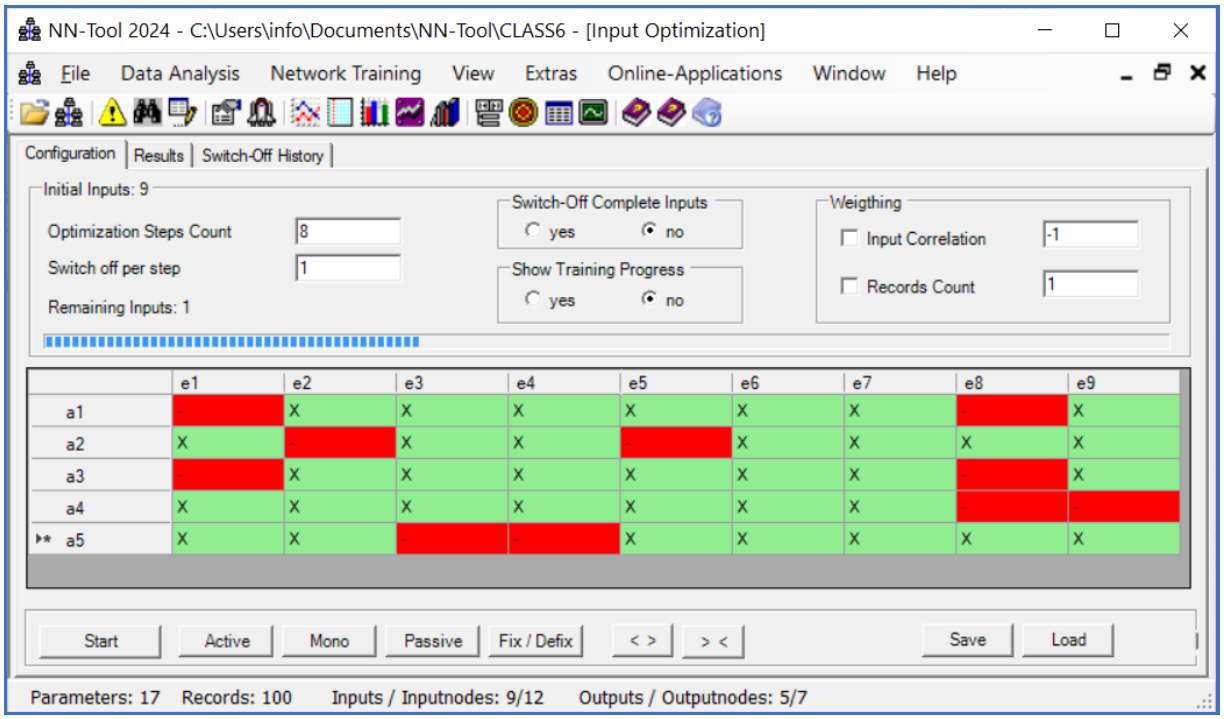
First load the "class6" network that was previously created using cross-validation and multithreading and select "**Network training \ Input Optimization**":



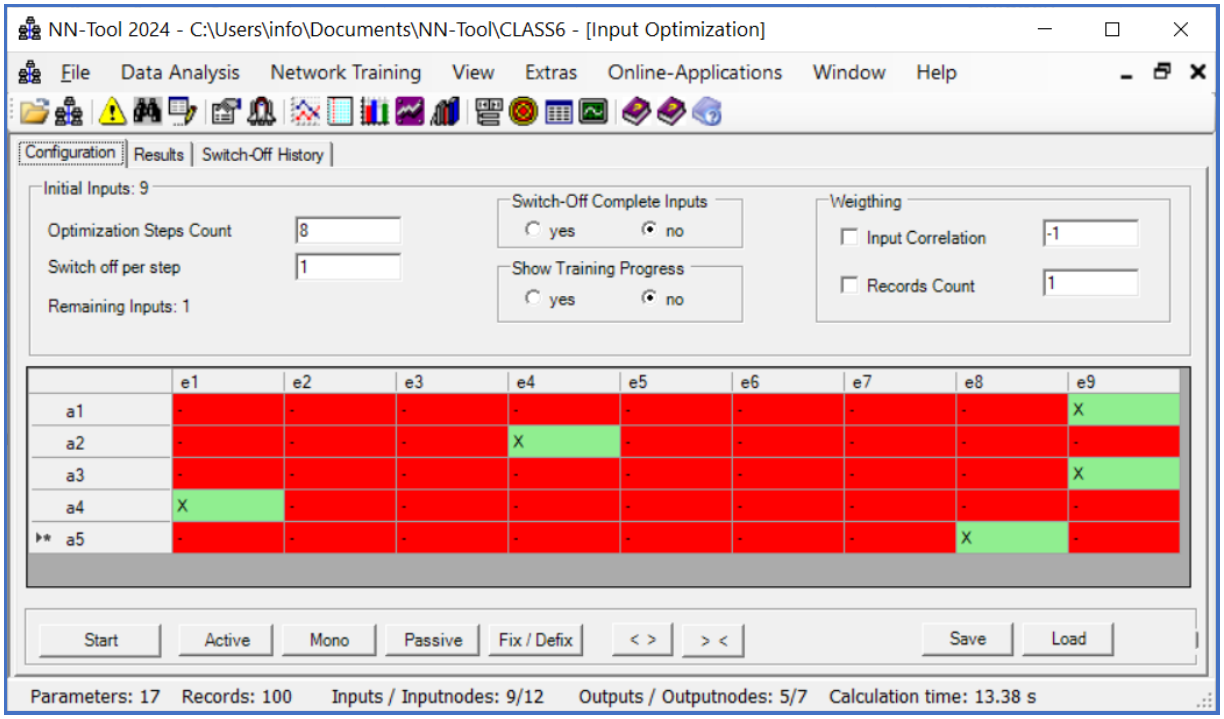
Method:
The user first specifies a number of optimization steps (e.g. 8) and the inputs to be switched off per step (e.g. 1). The values shown here are the defaults, namely with n inputs (n-1) steps in which only one input is switched off at a time. Other settings make particular sense if there is a large number of potential inputs for large number of data sets (computation time). For example, with hundred inputs you could carry out 33 runs with 3 inputs to be switched off each. It is important that at least one input remains at the end. **It is recommended to carry out the learning process using "crossvalidation" and "multithreading" (set training parameters accordingly in advance).**
After pressing the **Start** button, a 0-th training run takes place, in which all inputs actively affect all outputs (unless the parameters are switched completely passive). The optimization runs then take place (in the example 8). The following program is processed automatically for each optimization run:

1. Based on the previous run, the influences of the inputs on the outputs are calculated.
2. The inputs with the lowest **absolute** influences per output from the previous run are set passively (here 1 each).
3. A new network is created with the new input-output configuration. The setting under "Set learning parameters" applies.
4. The network quality of the configuration is determined and saved.

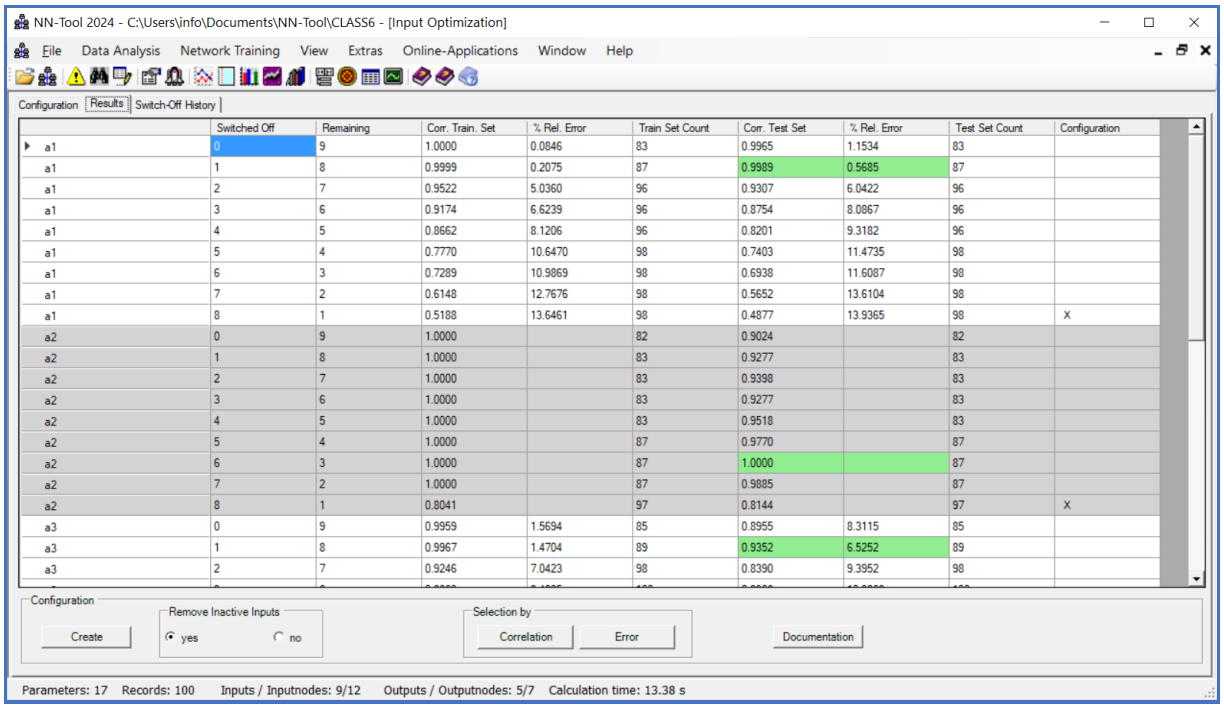
Intermediately:



After the last run, the network with the fewest inputs results (here only one input per output):



With each run, the number of active inputs decreases. After all runs (note: because of the 0-th run there is one more than selected) the results for all outputs and all configurations are displayed:

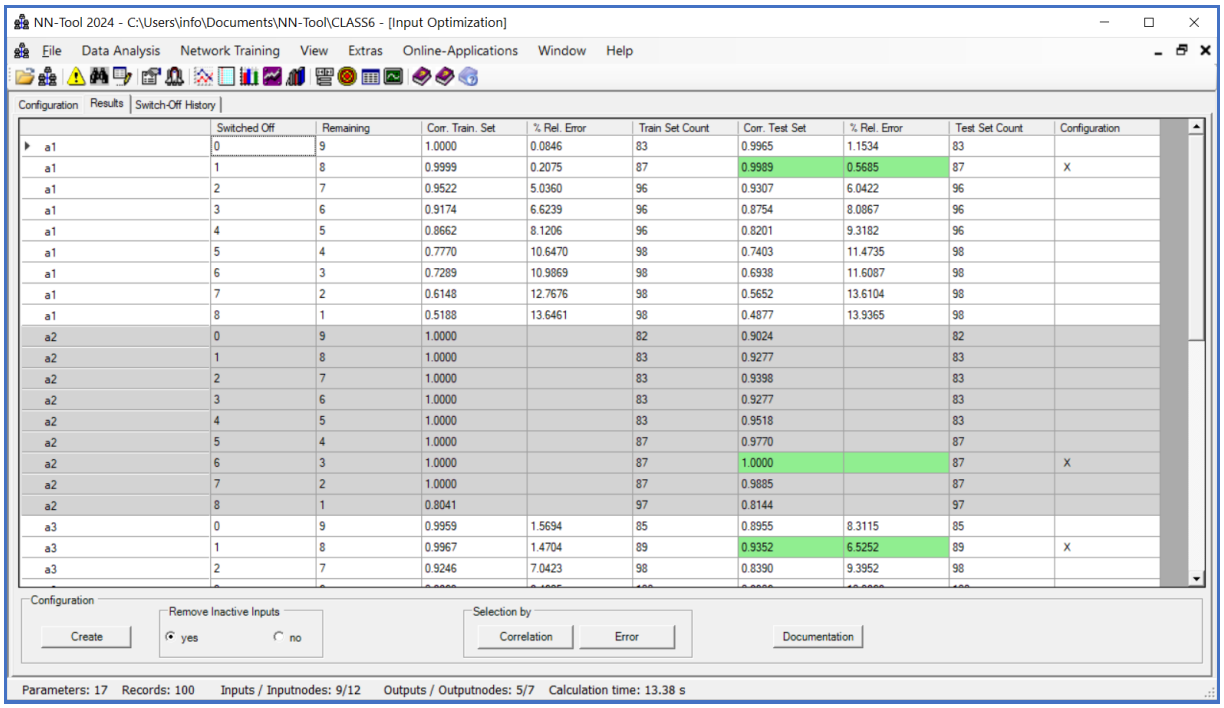


Attention: The network created is now the one with the fewest inputs. The corresponding configuration is marked by an X in the "Configuration" column. However, in general this is not the optimal configuration (especially not for every output).

Using the mouse (by clicking in the Configuration column), a **desired configuration** can now be selected for each output. Alternatively, the optimal configuration for all outputs can be selected by pressing the **Correlation or Error** buttons. **This configuration is then created by pressing the Create button**. The entire process can be documented using the **Documentation** button.

Further action:

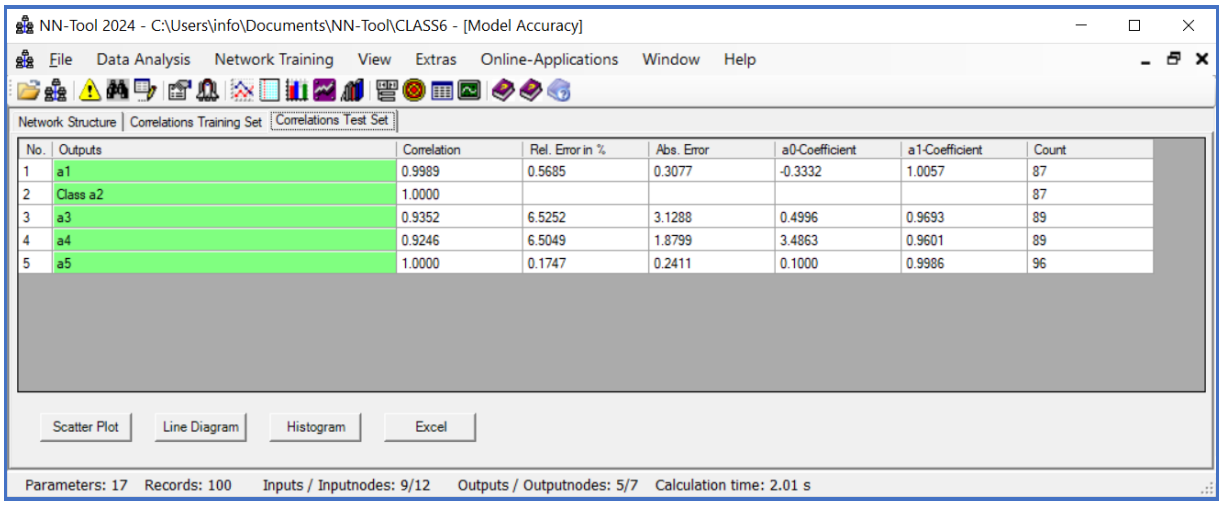
a) First press the "**Error**" button:



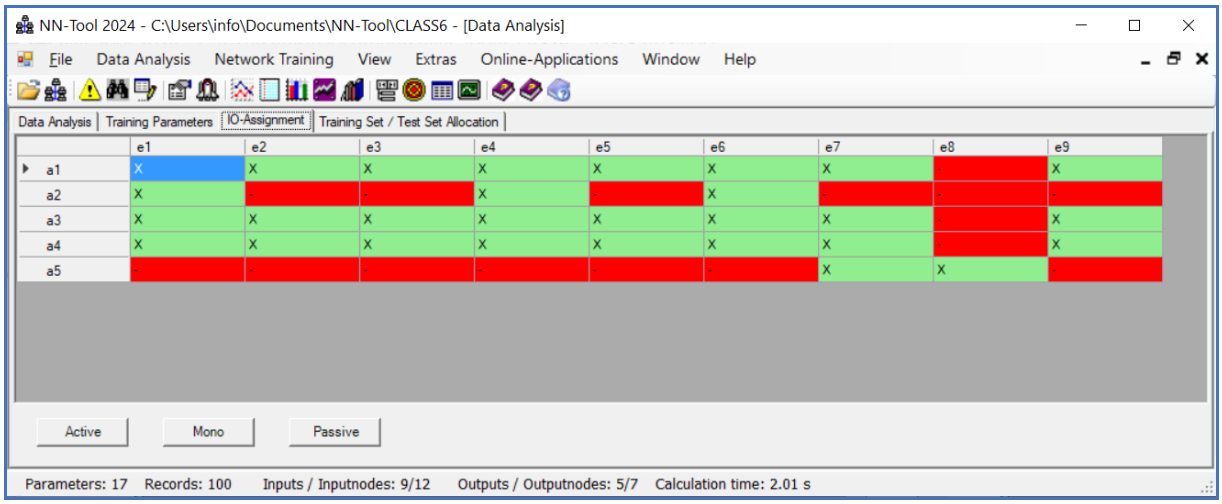
This means that the optimal configuration with regard to the error is selected for each output, which can be seen in the X positions.

b) Then press the "**Documentation**" button. An Excel folder is created that records the entire process.

c) Finally, press the "**Create**" button. Only then will the selected configuration be created.



The result is a network that is significantly improved in terms of model accuracy. The input-output assignment shows the configuration created:



Finally, the "**Automatic Documentation**" option is recommended again.

Hints:

1) **Common Error:** It is easy to press the Create button without first selecting the optimal network configuration. In this case, the network is generated with the minimum number of inputs. This generally shows very poor model accuracy. Then the input-output assignment must be reset and the entire process repeated.

2) **If the input parameters are incomplete**, the number of data sets available per output can change due to the sequential switching off of inputs during the process. This may also change the learning and test set allocations. In order to obtain statistically valid results in these cases, **the use of cross-validation is recommended.**

Additional options:

- **Switch-Off History tab**: The table indicates in which step of the process the respective input parameter was switched off.

- **Switch-Off Complete Inputs:** With this option, those inputs whose maximum influence on all outputs is the lowest are determined and then switched off completely for all outputs.

- **Show Training Progress**: If you select the " Show Training Progress no" option, the NN-Tool learning algorithm runs invisibly in the background. The progress bar is (for technical reasons) only updated after a complete learning process. So it may take a while.

- **Weighting:** By setting the two possible options "Input Correlation" and "Records Count", a weighted criterion is applied to determine the inputs to be switched off. The following evaluation formula applies to the i-th input parameter:

  $$P(i) = 10 \cdot AI(i) + \alpha \cdot MC(i) + \beta \cdot RDC(i)$$

  The performance P of the i-th input results from the **10-fold weighted** absolute influence "**AI**" of the input on the output currently being viewed (or on all outputs with the "Switch-Off Complete Inputs" option) and the penalty terms „**MC**" (= maximum correlation of the

i-th input to the other active inputs) and "**RDC**" (= relative data count = number of occurrences of the i-th input / total number of data records), weighted with the parameters α and β. Since the lowest possible correlation between the inputs is desired, the parameter α should of course be chosen to be negative. These two values can be specified in the user interface. If the corresponding check mark is not set, the associated parameter is set to zero. Then the inputs are switched off based only on their influences.

In the next chapter we will look at a special model structure that is particularly interesting for the topic of "**Predictive Maintenance**".

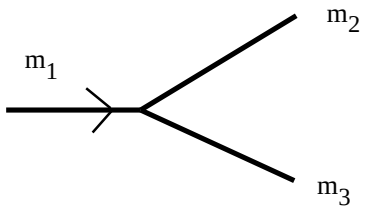
7. Data Reconciliation and Predictive Maintenance

This NN tool component makes it possible to automatically monitor a large number of measuring points in a process or large-scale system. The following goals should be achieved:

- **Monitoring of all relevant measuring points of processes / systems / subsystems.**
- **Data Reconciliation: Correction of measurement errors.**
- **Calculation of missing measurements.**
- **Predictive maintenance: early detection of impermissible operating states.**
- **As little engineering effort as possible.**

Method: Combination of neural networks and principal component analysis (nonlinear PCA). The essential approach is based on the usually strong redundancy of the measured values.

Simple example: At a branch of the following form



the formula $m_1 = m_2 + m_3$ applies to the mass flows. The physical quantities m1, m2, m3 lie in the 2-dimensional area (subspace) defined by $m_3 = m_1 - m_2$. So we have 3 measurements with only 2 degrees of freedom. These surplus (redundant) measurements can be used for measurement data monitoring (Data Reconciliation).

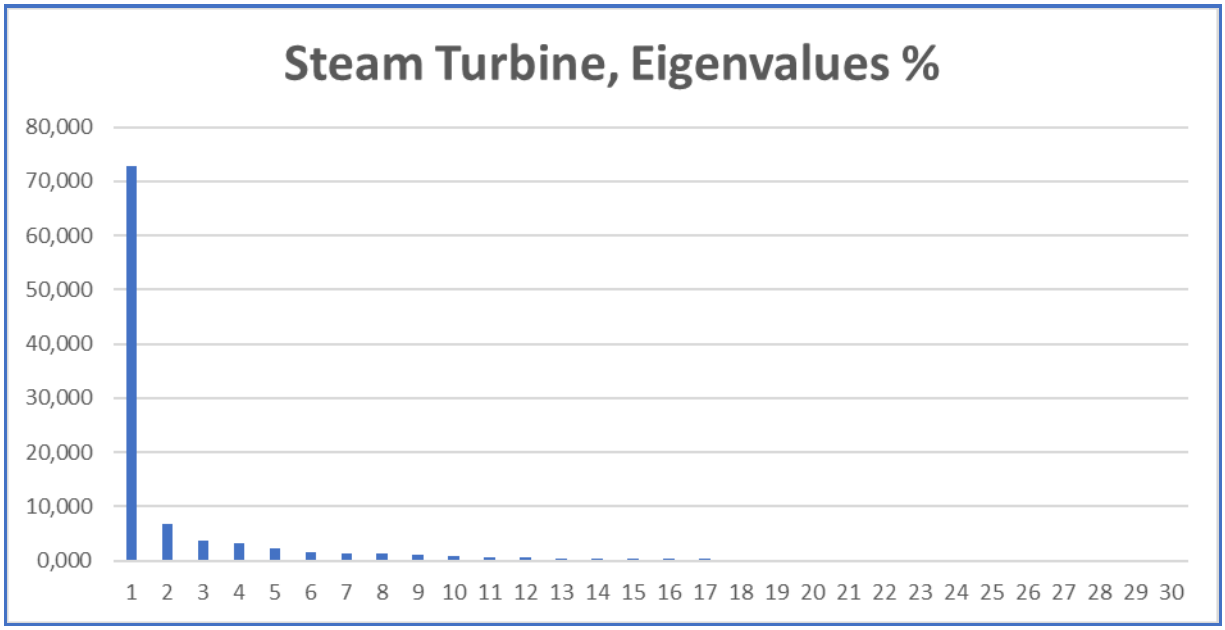
In a **large-scale plant** you usually have the following situation:

- Numerous redundant measurement variables.
- Existing physical relationships reduce the degrees of freedom.
- Number of degrees of freedom f is significantly smaller than the number of measured variables.
- Process status already clearly determined by f measured values.
- Use of additional measured values for measurement data monitoring.
- But model required:
  – Physical model
  – Statistical model from measurement data

We now want to automatically create a measurement data monitoring application ("data reconciliation application") based on measurement data with the least possible engineering effort. The approach consists of a combination of principal component analysis (PCA = Principal Component Analysis) with neural networks. The starting point is the so-called **correlation matrix** (see also appendix: Special functions for data analysis and preprocessing). The following applies:

- **Correlation matrix** consists of the correlation coefficients of each measurement variable with each other.
- **Principal component analysis (PCA)**: Determination of the eigenvalues and eigenvectors of the correlation matrix (linear algebra).
- **Eigenvalues** of the correlation matrix are the key to determining the number of **degrees of freedom** of the process.
- **Eigenvectors** for the largest eigenvalues form (linear) optimal state variables of the process.
- **Measurement data monitoring** results from the **back calculation** of all measured variables from the eigenvectors.
- Back calculation can be done linearly (classic PCA) or non-linearly using neural networks.

We use PCA to determine the eigenvalues of the correlation matrix. The eigenvalues contain the essential information about the degrees of freedom of the system, i.e. how many variables can really be specified independently. The following diagram shows the situation with a steam turbine:



- More than 60 measurement variables. So there are correspondingly many eigenvalues. The diagram shows the progression for the 30 largest eigenvalues.

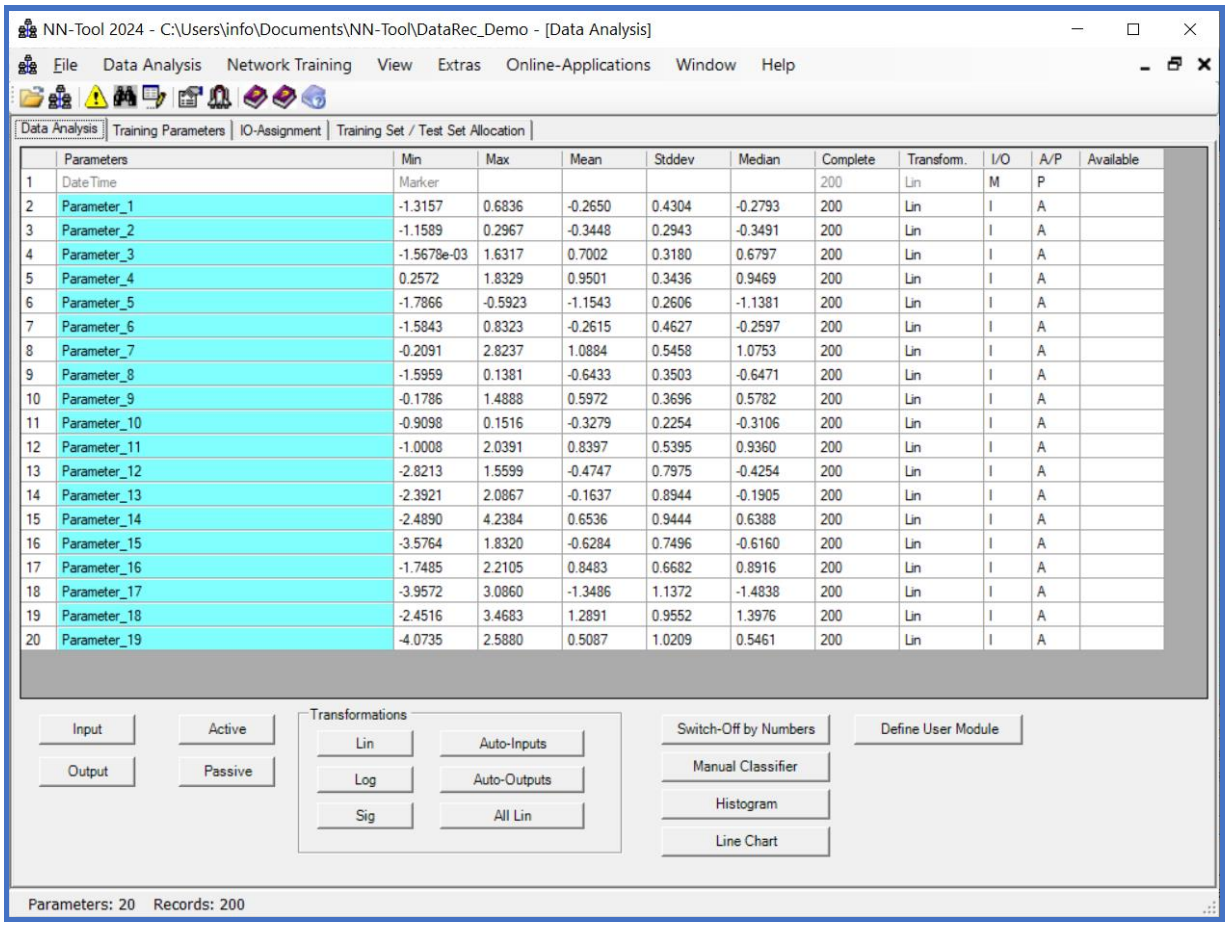
- But only about a dozen relevant eigenvalues/degrees of freedom. The first 11 eigenvalues already contain more than 95% of the total eigenvalue strength.
- The remaining eigenvalues essentially only represent the measurement noise. The corresponding components can be neglected.
- The measured variables can then be reconstructed from the eigenvectors to the relevant eigenvalues. This means that the system is already determined by specifying 11 values.

Method:

1. Load and analyze file with process data.
2. Calculate correlation matrix.
3. Perform PCA.
4. Determine number of relevant eigenvalues (degrees of freedom).
5. Compute transformations to the eigenvectors (and back).
6. Expand process data to include coordinates in the eigenvectors.
7. Create neural network model from the coordinates to the measurement data.
8. Link transformations and network model to the finished measurement data monitoring application.
9. Couple application with process control system.

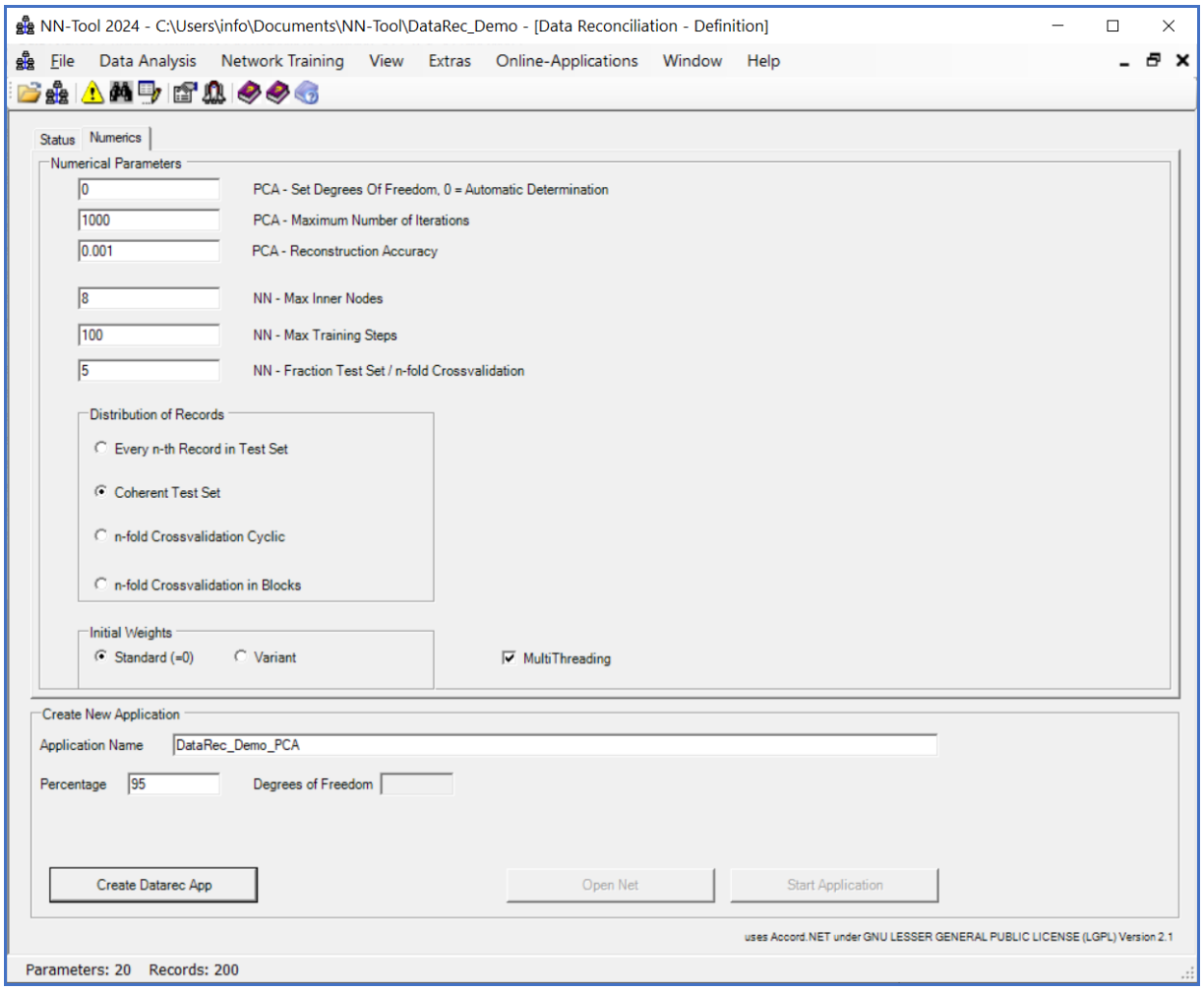
Points 1. to 8. can now be carried out automatically using the NN tool component Data Reconciliation. A corresponding library is available for coupling (point 9.). The procedure is explained below using the example file "**DataRec_Demo.xls**" in the NN-Tool directory. The Excel folder contains the spreadsheets "**DataRec_Demo**" and "**DataRec_Error_2**".

- First create the corresponding .pat files.
- Now open **DataRec_Demo.pat**, set the "**Date Time**" parameter as an marker (important!!), and carry out the analysis:



	Parameters	Min	Max	Mean	Stddev	Median	Complete	Transform.	I/O	A/P	Available
1	DateTime	Marker					200	Lin	M	P	
2	Parameter_1	-1.3157	0.6836	-0.2650	0.4304	-0.2793	200	Lin	I	A	
3	Parameter_2	-1.1589	0.2967	-0.3448	0.2943	-0.3491	200	Lin	I	A	
4	Parameter_3	-1.5678e-03	1.6317	0.7002	0.3180	0.6797	200	Lin	I	A	
5	Parameter_4	0.2572	1.8329	0.9501	0.3436	0.9469	200	Lin	I	A	
6	Parameter_5	-1.7866	-0.5923	-1.1543	0.2606	-1.1381	200	Lin	I	A	
7	Parameter_6	-1.5843	0.8323	-0.2615	0.4627	-0.2597	200	Lin	I	A	
8	Parameter_7	-0.2091	2.8237	1.0884	0.5458	1.0753	200	Lin	I	A	
9	Parameter_8	-1.5959	0.1381	-0.6433	0.3503	-0.6471	200	Lin	I	A	
10	Parameter_9	-0.1786	1.4888	0.5972	0.3696	0.5782	200	Lin	I	A	
11	Parameter_10	-0.9098	0.1516	-0.3279	0.2254	-0.3106	200	Lin	I	A	
12	Parameter_11	-1.0008	2.0391	0.8397	0.5395	0.9360	200	Lin	I	A	
13	Parameter_12	-2.8213	1.5599	-0.4747	0.7975	-0.4254	200	Lin	I	A	
14	Parameter_13	-2.3921	2.0867	-0.1637	0.8944	-0.1905	200	Lin	I	A	
15	Parameter_14	-2.4890	4.2384	0.6536	0.9444	0.6388	200	Lin	I	A	
16	Parameter_15	-3.5764	1.8320	-0.6284	0.7496	-0.6160	200	Lin	I	A	
17	Parameter_16	-1.7485	2.2105	0.8483	0.6682	0.8916	200	Lin	I	A	
18	Parameter_17	-3.9572	3.0860	-1.3486	1.1372	-1.4838	200	Lin	I	A	
19	Parameter_18	-2.4516	3.4683	1.2891	0.9552	1.3976	200	Lin	I	A	
20	Parameter_19	-4.0735	2.5880	0.5087	1.0209	0.5461	200	Lin	I	A	

- The application contains 19 numerical parameters between which mathematical relationships exist. Which? How many? Initially unclear.
- Now start the “Data Reconciliation” item in the “Network Training” main menu. All active numerical parameters are included (here 19). Classifiers can currently not be taken into account. The following window appears:



Here you can first set some numerical parameters of the method:

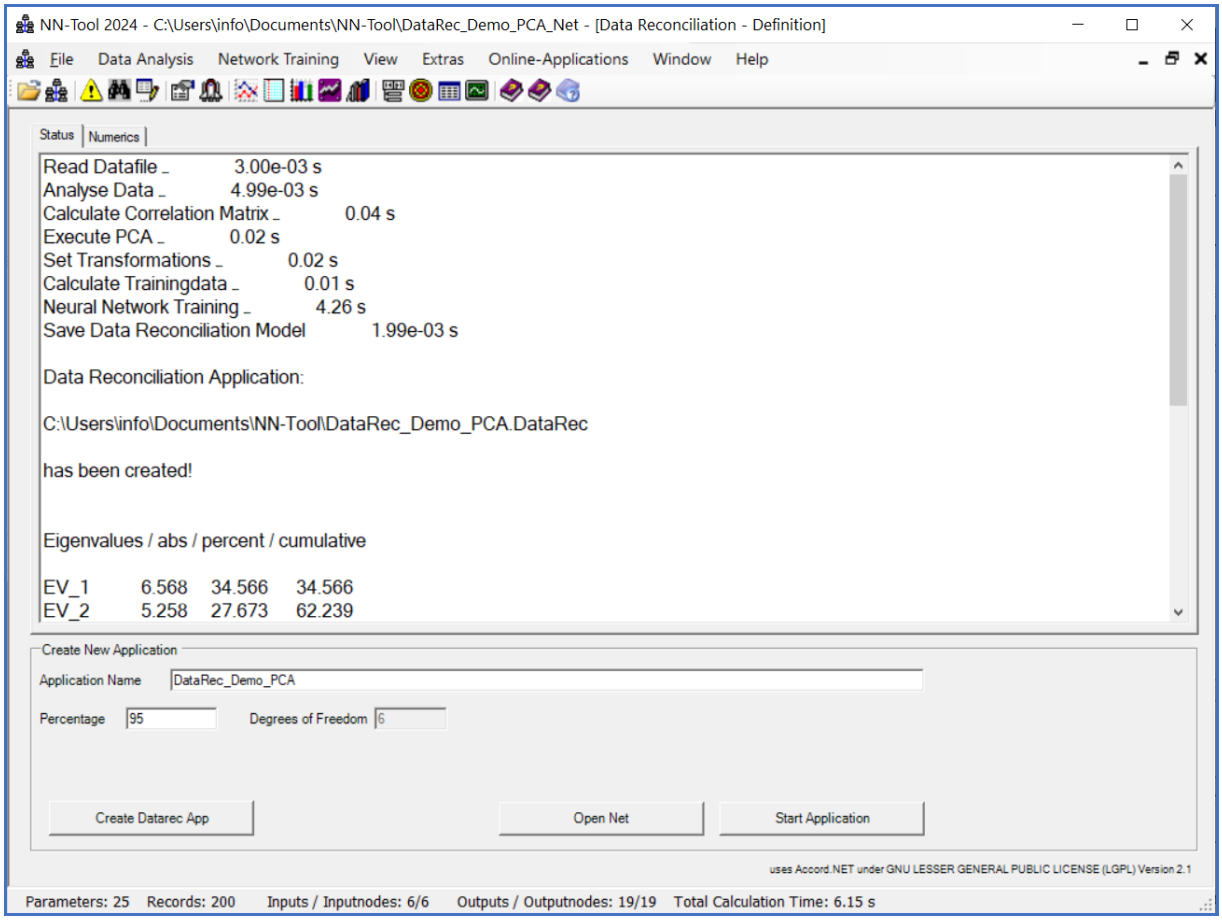
- First you have to decide whether you want to set the number of degrees of freedom directly ("**PCA – Set degrees of freedom**" option) or whether the number of degrees of freedom should be determined automatically based on the **specified percentage**. A **percentage of 95%** (as in the example) means: "Consider so many eigenvalues that the summed first f many eigenvalues exceed 95% of the total sum of the eigenvalues". This option is recommended.

- The two options "**PCA – Maximum Number of Iterations**" and "**PCA – Reconstruction Accuracy**" regulate the procedure for calculating the principal components if a data set is incomplete. In this case, the calculation can still be carried out as long as there are more measured values in a data set than the number of degrees of freedom f. Example: 19 measuring points with 6 degrees of freedom means that a single data set can still be monitored if it still has at least 7 measured variables, i.e. the set may have a maximum of 12 missing measured values. However, in the case of missing measured variables, the calculation requires an iterative numerical procedure, which is controlled by the parameters mentioned.

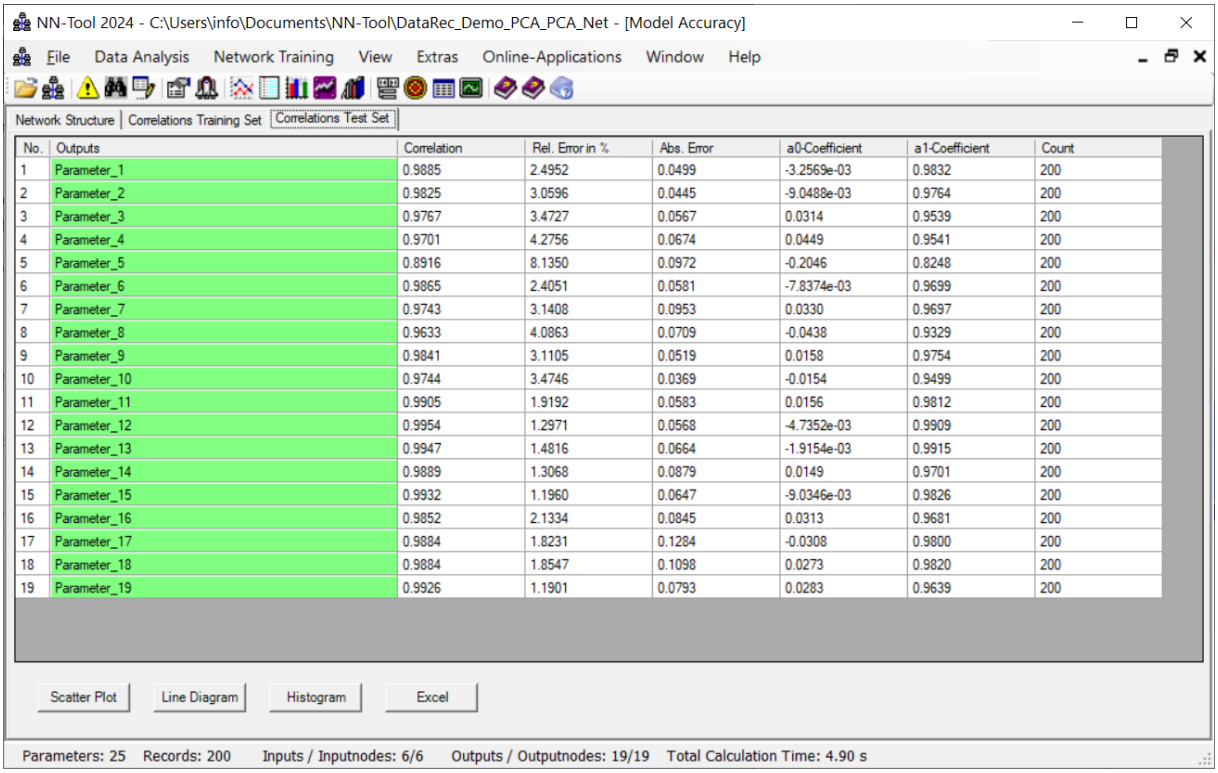
- The other parameters then control the **learning algorithm** of the neural network in the usual way. Since the data in typical applications are usually consecutive measurements,

the options "**Coherent Test Set**" or, if the data sets are not too large, the option "**n-fold Crossvalidation in Blocks**" are recommended.

- Select the "**Crossvalidation n-fold blockwise**" option and press the "**Create Datarec App**" button. The system then carries out the necessary procedural steps

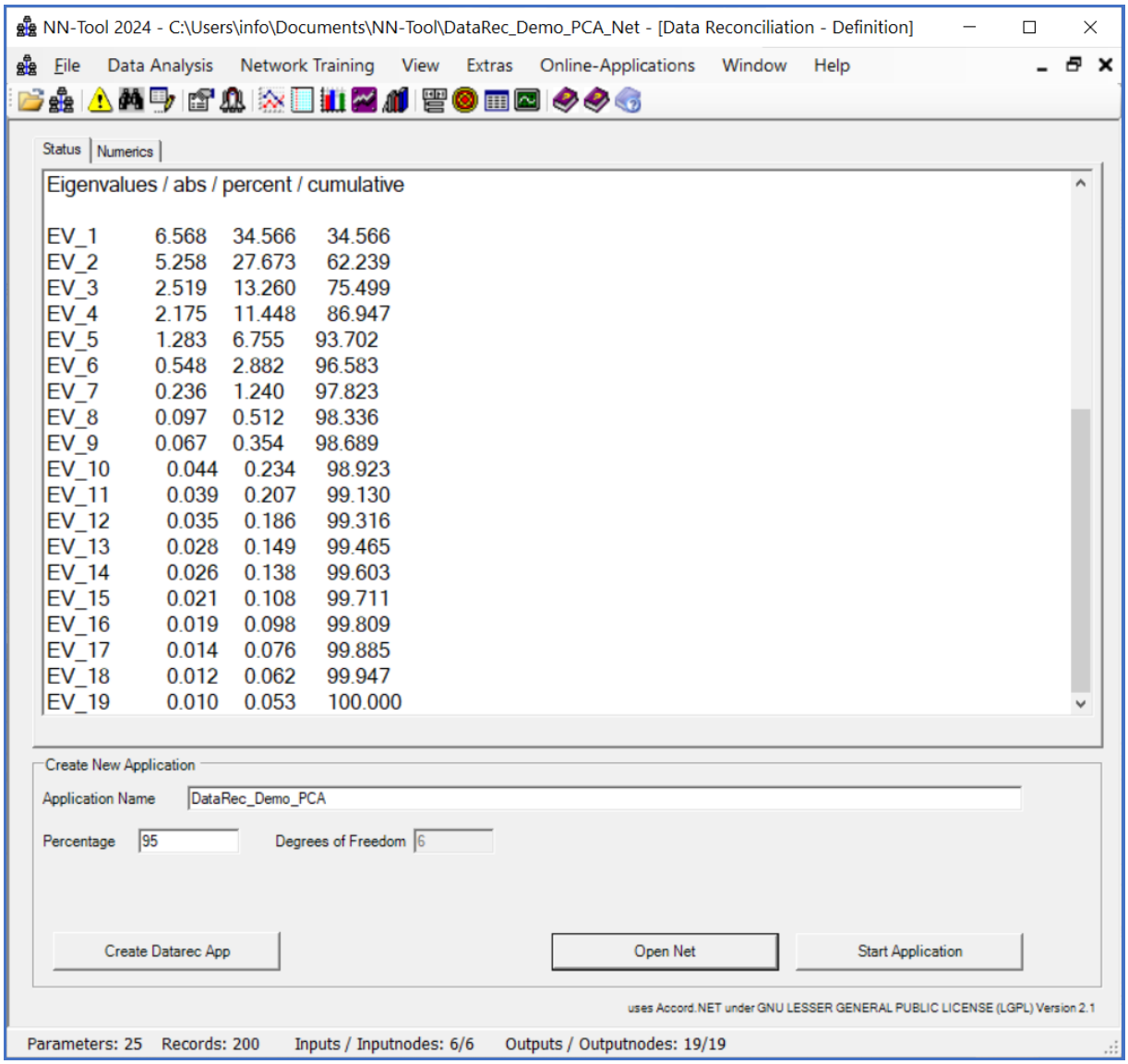


and then displays the determined number of degrees of freedom (here 6) as well as the achieved model accuracy:



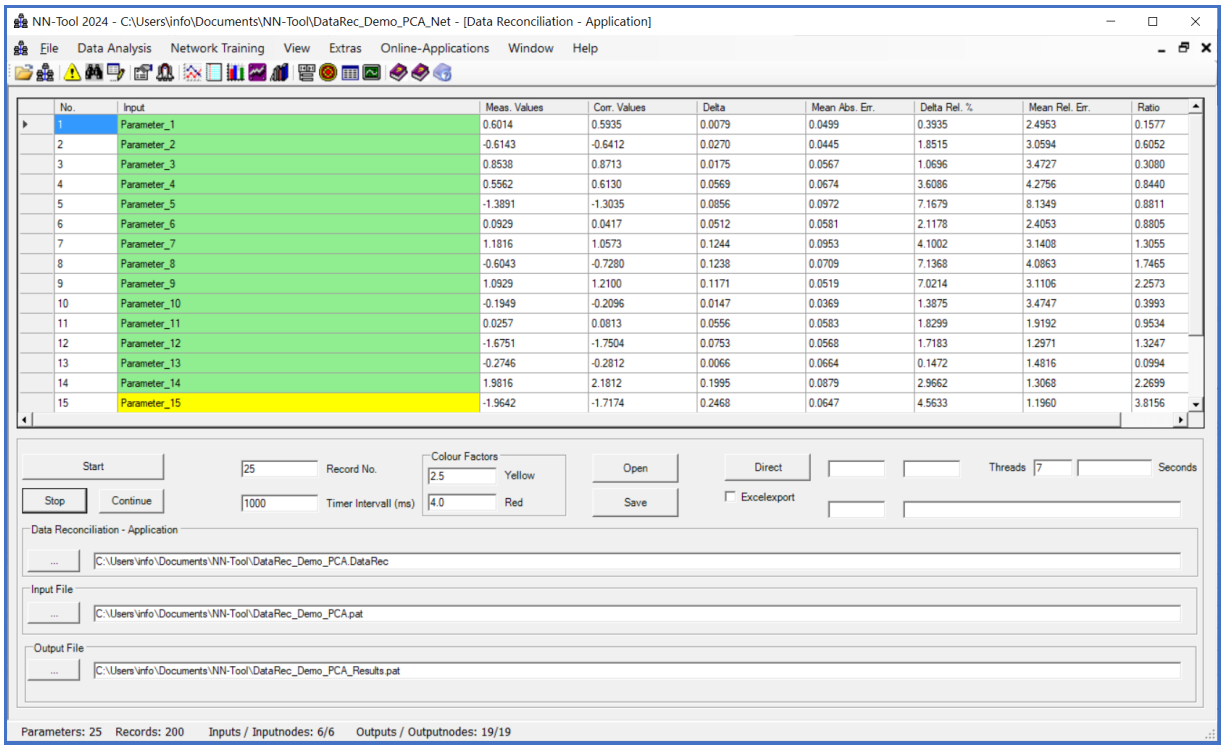
No.	Outputs	Correlation	Rel. Error in %	Abs. Error	a0-Coefficient	a1-Coefficient	Count
1	Parameter_1	0.9885	2.4952	0.0499	-3.2569e-03	0.9832	200
2	Parameter_2	0.9825	3.0596	0.0445	-9.0488e-03	0.9764	200
3	Parameter_3	0.9767	3.4727	0.0567	0.0314	0.9539	200
4	Parameter_4	0.9701	4.2756	0.0674	0.0449	0.9541	200
5	Parameter_5	0.8916	8.1350	0.0972	-0.2046	0.8248	200
6	Parameter_6	0.9865	2.4051	0.0581	-7.8374e-03	0.9699	200
7	Parameter_7	0.9743	3.1408	0.0953	0.0330	0.9697	200
8	Parameter_8	0.9633	4.0863	0.0709	-0.0438	0.9329	200
9	Parameter_9	0.9841	3.1105	0.0519	0.0158	0.9754	200
10	Parameter_10	0.9744	3.4746	0.0369	-0.0154	0.9499	200
11	Parameter_11	0.9905	1.9192	0.0583	0.0156	0.9812	200
12	Parameter_12	0.9954	1.2971	0.0568	-4.7352e-03	0.9909	200
13	Parameter_13	0.9947	1.4816	0.0664	-1.9154e-03	0.9915	200
14	Parameter_14	0.9889	1.3068	0.0879	0.0149	0.9701	200
15	Parameter_15	0.9932	1.1960	0.0647	-9.0346e-03	0.9826	200
16	Parameter_16	0.9852	2.1334	0.0845	0.0313	0.9681	200
17	Parameter_17	0.9884	1.8231	0.1284	-0.0308	0.9800	200
18	Parameter_18	0.9884	1.8547	0.1098	0.0273	0.9820	200
19	Parameter_19	0.9926	1.1901	0.0793	0.0283	0.9639	200

Close the model accuracy window (you can open it again at any time with “Open Net”):

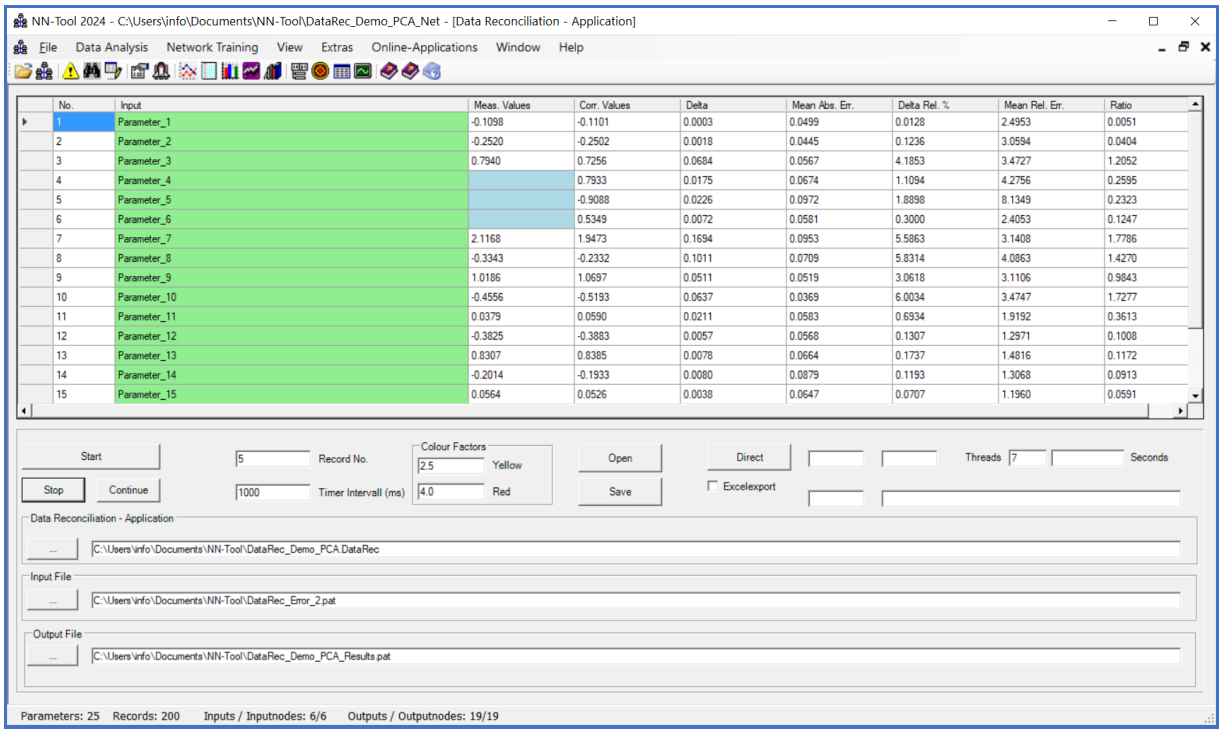


As you can see from the illustration, the first 6 eigenvalues are sufficient to exceed the 95% mark.

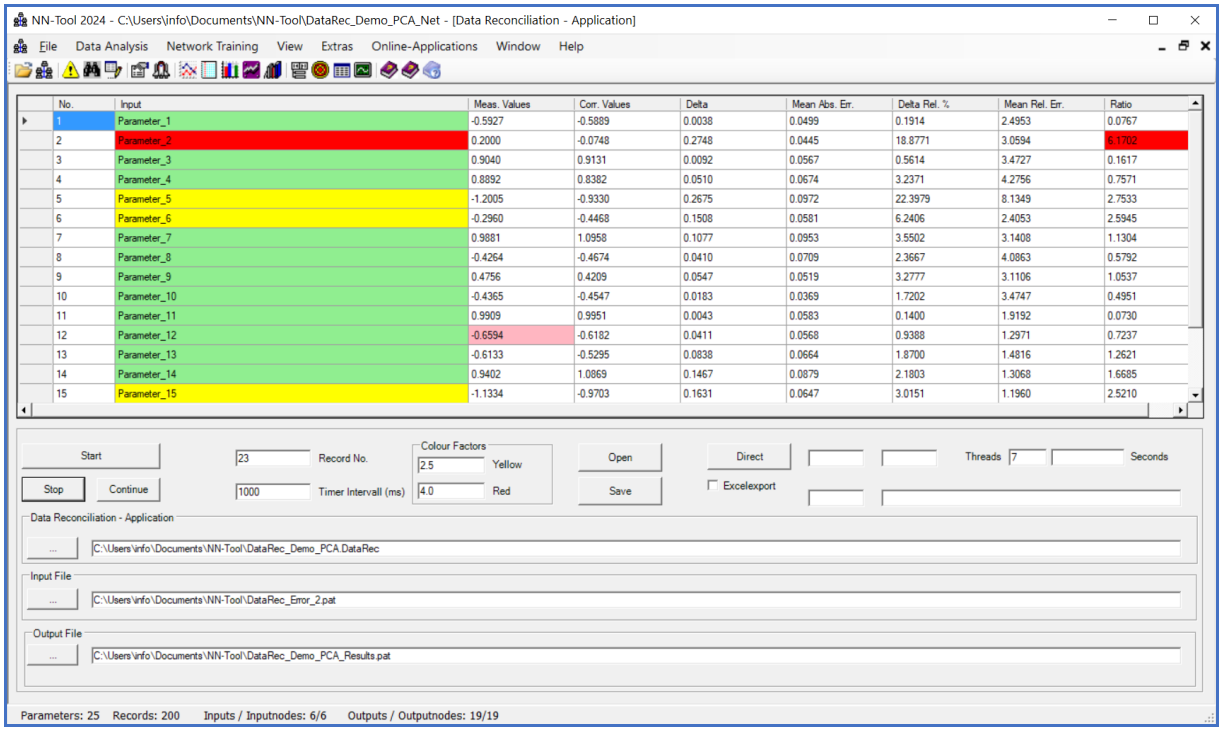
Now press the "**Start Application**" button and press "**Start**" in the new window:



The data records are now successively read in from the "**Input File**" and corrected (columns "**Meas. Values**" and "**Corr. Values**") using the **Data Reconciliation Application** "C:\Users\baer\Documents\NN-Tool\DataRec_Demo_PCA.DataRec" that has just been created. The "**Delta**" column measures the difference. If the difference is too large in relation to the model error, it is marked in color (here yellow). Since only the training data is initially pushed through the system, there are hardly any deviations. Stop the process and now select the "**DataRec_Error_2.pat**" file created at the beginning as the Input File and then start again:



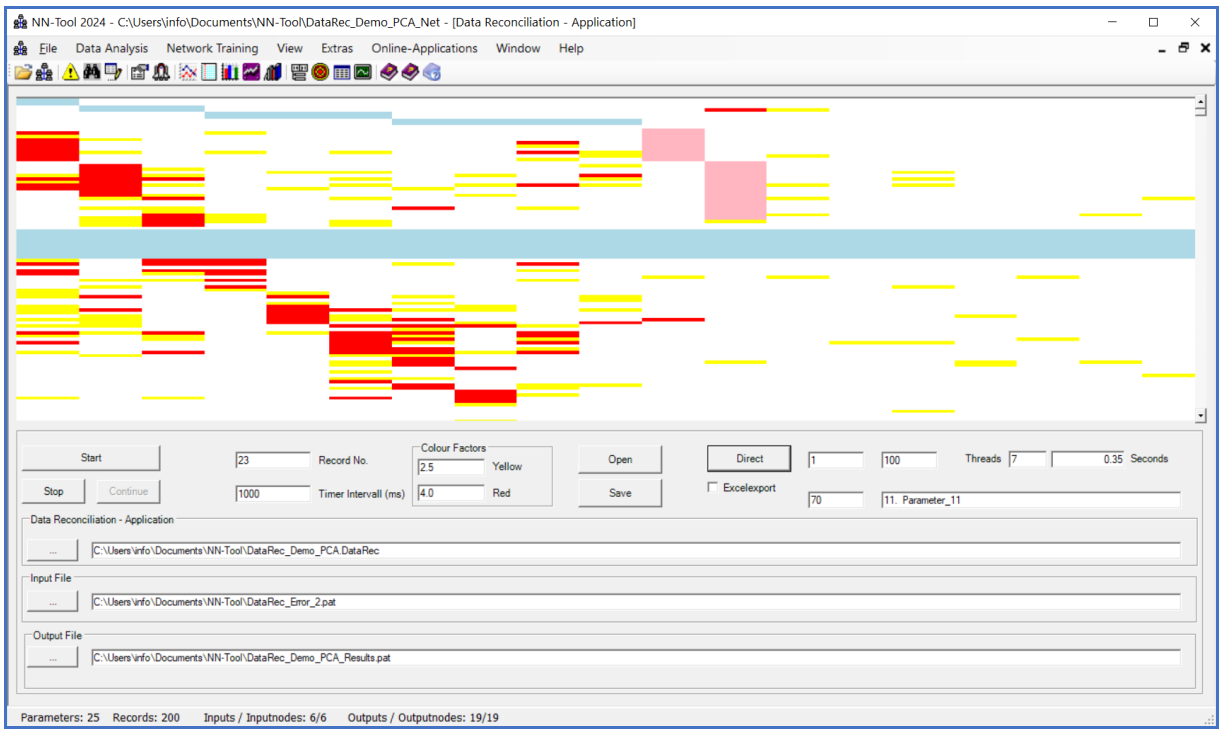
The blue highlighted cells in the Measured column indicate that the corresponding measurement values are not available. From data set 11 onwards, massive measurement errors occur in this input file:



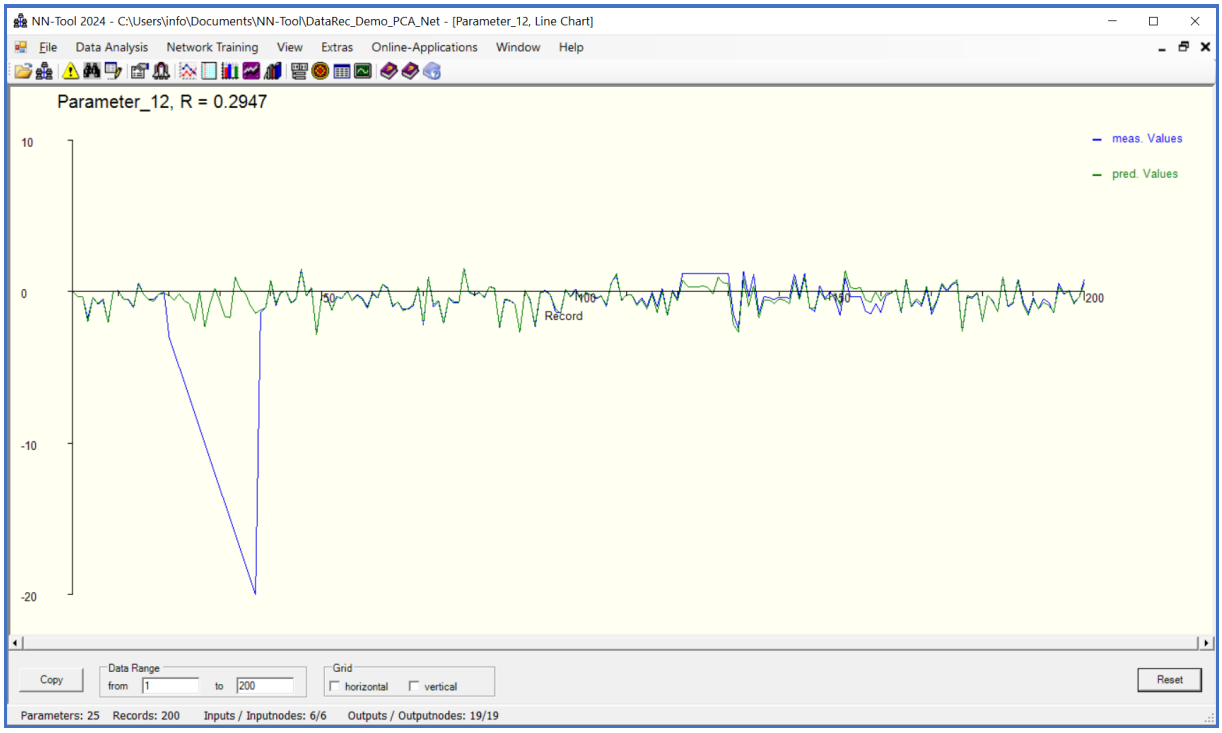
No.	Input	Meas. Values	Corr. Values	Delta	Mean Abs. Err.	Delta Rel. %	Mean Rel. Err.	Ratio
1	Parameter_1	-0.5927	-0.5889	0.0038	0.0499	0.1914	2.4953	0.0767
2	Parameter_2	0.2000	-0.0748	0.2748	0.0445	18.8771	3.0594	6.1702
3	Parameter_3	0.9040	0.9131	0.0092	0.0567	0.5614	3.4727	0.1617
4	Parameter_4	0.8892	0.8382	0.0510	0.0674	3.2371	4.2756	0.7571
5	Parameter_5	-1.2005	-0.9330	0.2675	0.0972	22.3979	8.1349	2.7533
6	Parameter_6	-0.2960	-0.4468	0.1508	0.0581	6.2406	2.4053	2.5945
7	Parameter_7	0.9881	1.0958	0.1077	0.0953	3.5502	3.1408	1.1304
8	Parameter_8	-0.4264	-0.4674	0.0410	0.0709	2.3667	4.0863	0.5792
9	Parameter_9	0.4756	0.4209	0.0547	0.0519	3.2777	3.1106	1.0537
10	Parameter_10	-0.4365	-0.4547	0.0183	0.0369	1.7202	3.4747	0.4951
11	Parameter_11	0.9909	0.9951	0.0043	0.0583	0.1400	1.9192	0.0730
12	Parameter_12	-0.6594	-0.6182	0.0411	0.0568	0.9388	1.2971	0.7237
13	Parameter_13	-0.6133	-0.5295	0.0838	0.0664	1.8700	1.4816	1.2621
14	Parameter_14	0.9402	1.0869	0.1467	0.0879	2.1803	1.3068	1.6685
15	Parameter_15	-1.1334	-0.9703	0.1631	0.0647	3.0151	1.1960	2.5210

The color marking in the “Ratio” column marks the value that shows the greatest deviation (missing/reconstructed measured values excluded).

All data sets can be forecast at once using the “Direct” button. At the same time, the ratio values are shown in the following diagram:



Each line of the display corresponds to a data set, here in the range from 1 to 100. Further records are displayed using the scroll wheel. If you move the mouse into the diagram, the number of the corresponding data record (here number 70) and the parameter name are displayed (here 11th parameter called Parameter_11). The colors blue, yellow and red have the already known meanings, the color white corresponds to a measured value that does not deviate too much from the calculated value (ratio < yellow value, i.e. green according to the previous designation). Pink markings also appear. These indicate that the corresponding measured value has left its training range. Such a measurement can potentially severely distort the entire data validation. For this reason, it is not taken into account for data validation but is reconstructed. By double-clicking or right-clicking on the diagram, you can get the line chart of the corresponding parameter:



Here you can clearly see the effect of leaving the training range around record number 25.

Set the time interval to "1" and save the entire configuration under a file name of type **".DataRecApp**".
After restarting NN-Tool, you can call up the application again in the main menu **"Online-Applications"**, menu item "**Data Reconciliation – Application**".

A corresponding .NET library is available for integration into online systems (available upon request).

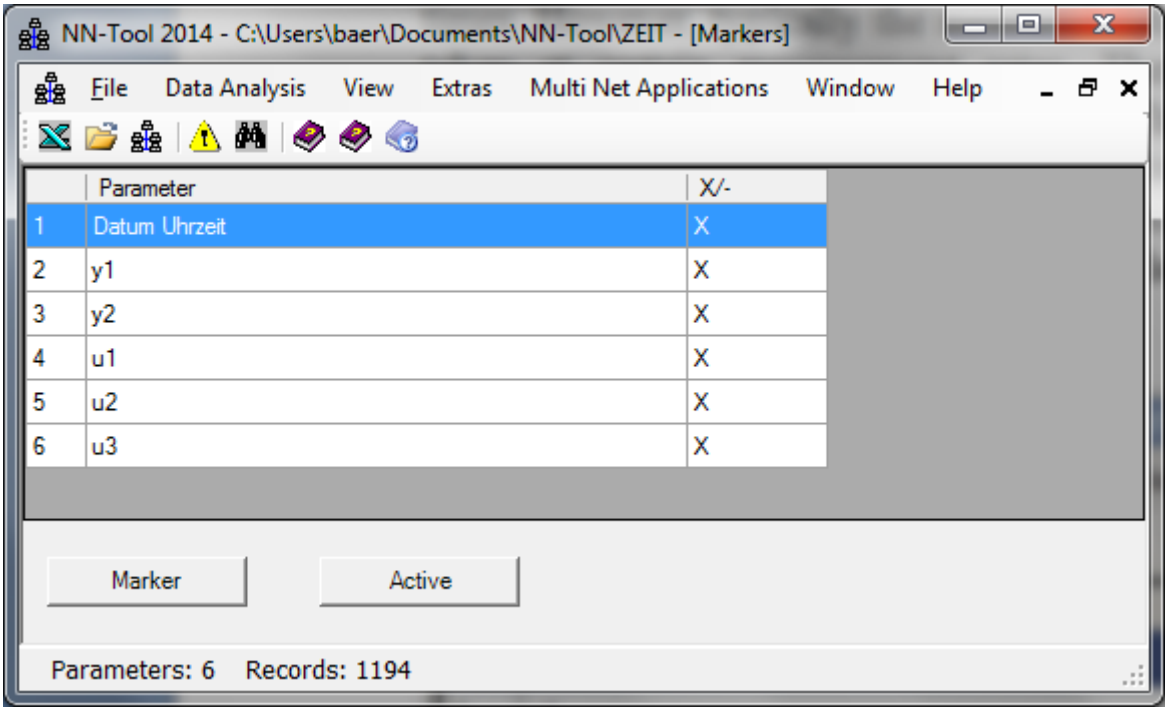
Hints:

- Currently the methodology can only be used for numerical parameters.
- If necessary, pre-filtering of the raw data is possible using the "**Data File Filter**" functionality in the "**File**" main menu (see Appendix 13).
- If markers (identifiers) occur, they should definitely be marked accordingly, otherwise the application module may cause inconsistencies with alternative test data sets.
- The Accord.Net library is used to carry out the principal component analysis. The use is

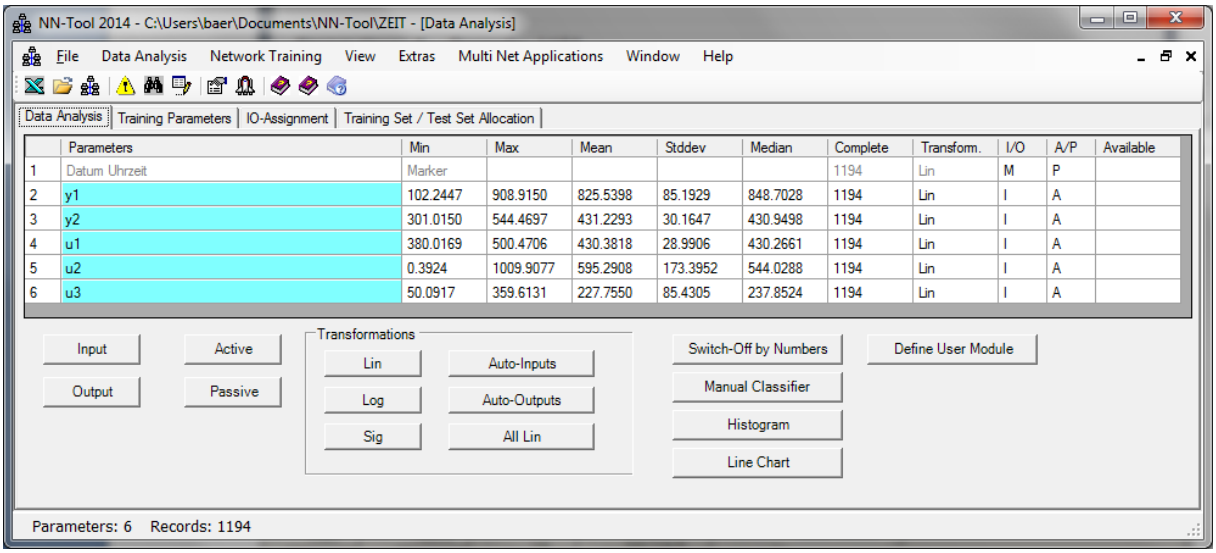
restricted to the conditions of the “GNU LESSER GENERAL PUBLIC LICENSE”. The conditions can be viewed in the appendix.

8. Time Series

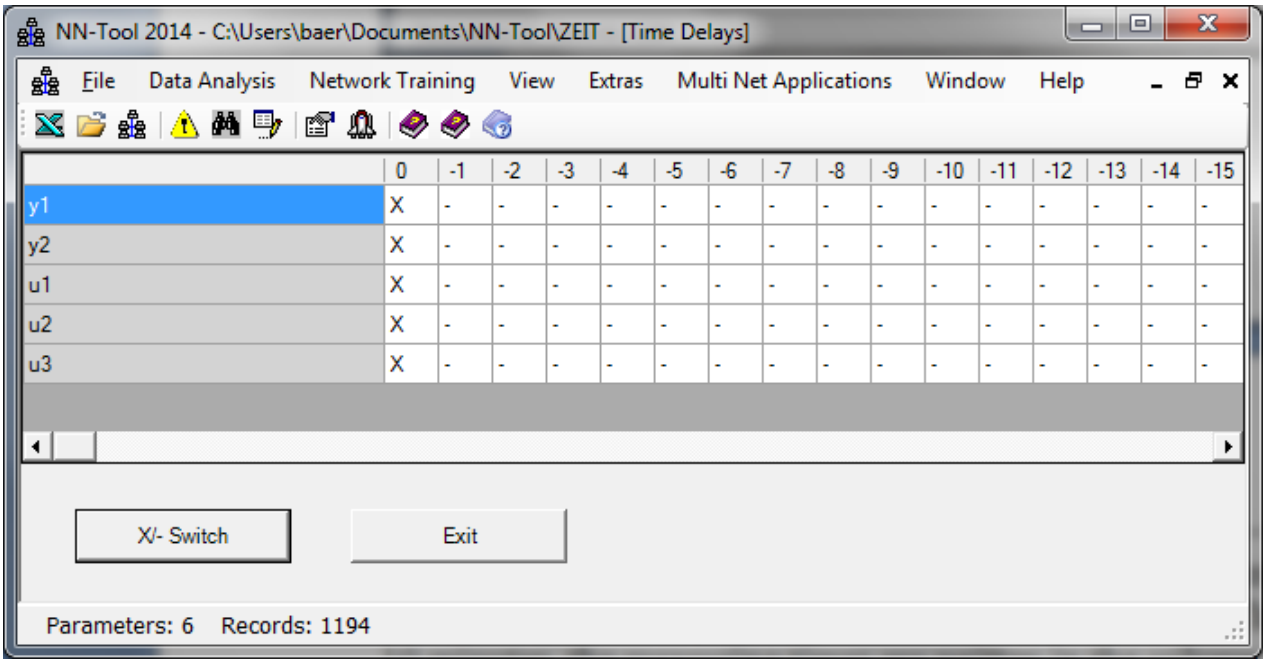
It is characteristic for modeling of time series that the models contain parameters at different times. However, typically the measured data are there in a form that the measured values are taken at certain measurement rates. Therefore the time delays occurring between the parameters have to be processed as well. The procedure is explained on the example Zeit.xls. Start NN-Tool, create and open the data file. The following is an indication that (for performance reasons) only the first 1000 records of the data are displayed.
Now choose the menu point **Deactivate Markers**:



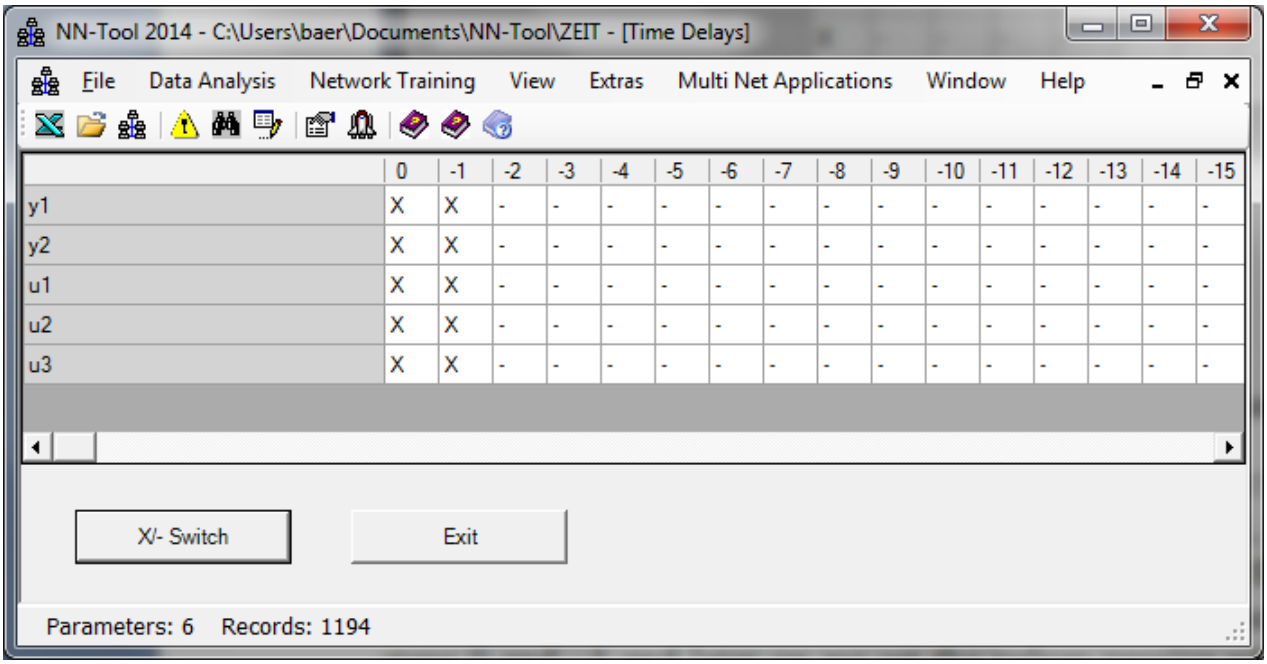
The first parameter “Datum Uhrzeit (Date Time) is of the type Date/Time, and so NN-Tool would treat it as a classifier with more than 1000 classes, what we do not want. Therefore deactivate this parameter and following execute the analysis:



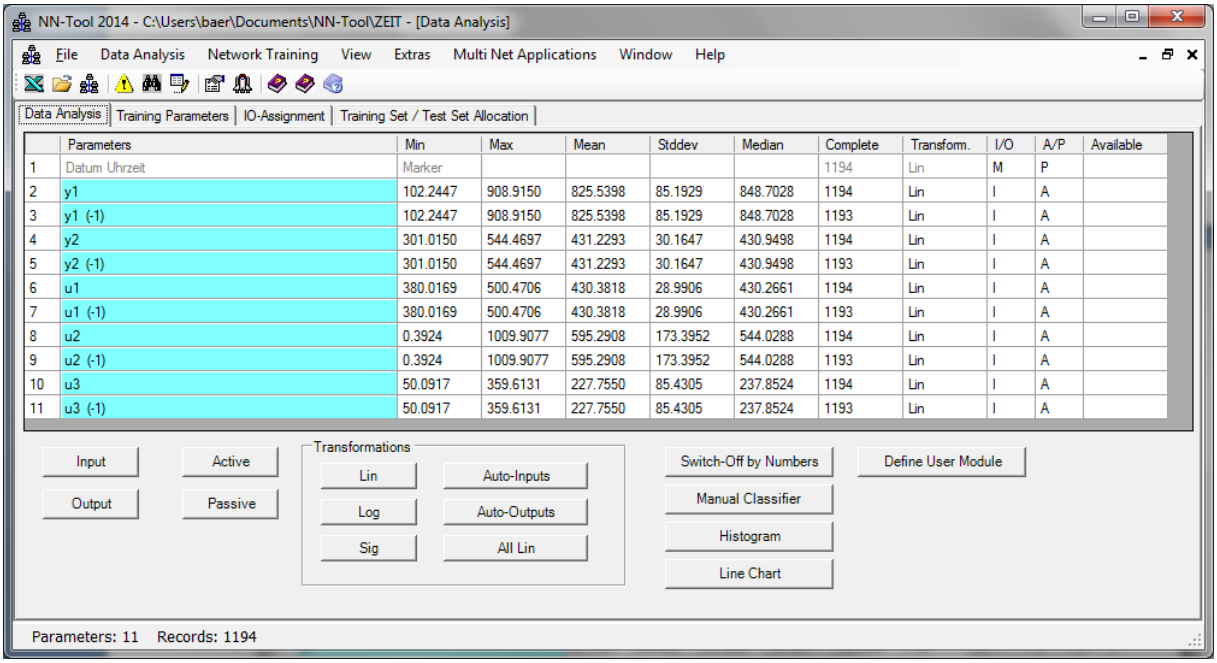
With this problem there is a connection between the values y1 and y2 at the present time with the values y1, y2, u1, u2, u3 at one time step in the past. The values have been measured every 10 minutes, the measuring times are written in the column **Datum Uhrzeit (Date Time)**. Now the right time delays have to be included.
Close the window and execute **Data Analysis/Time Delays**:



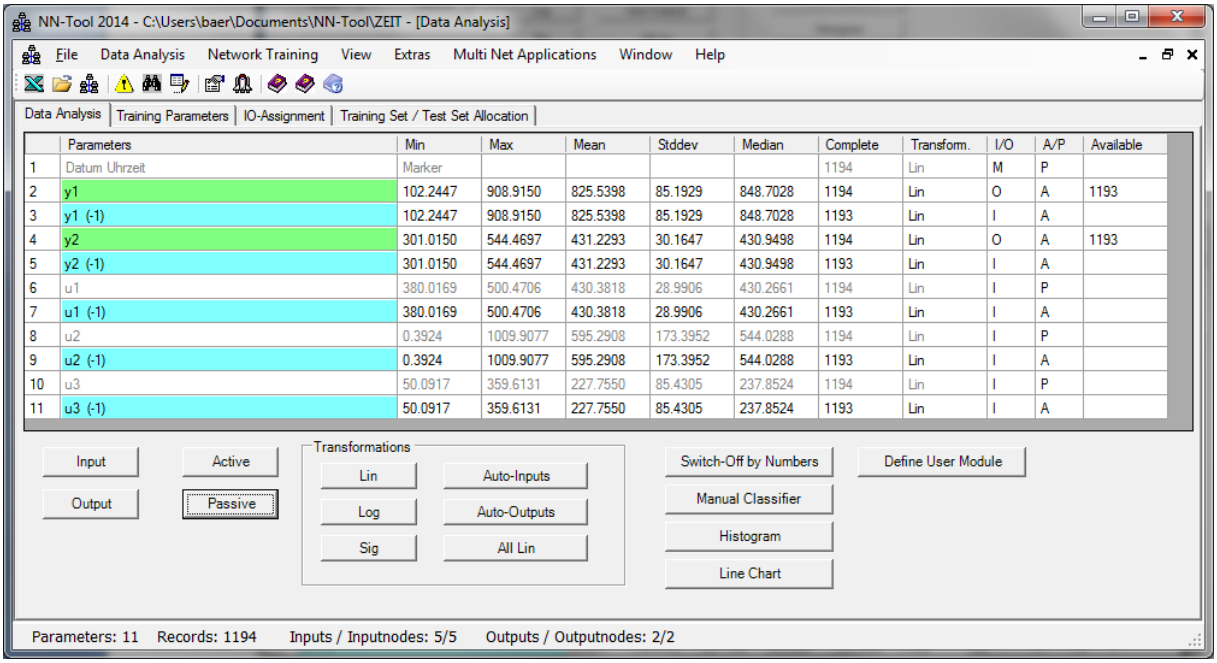
Determine the time steps by X/- Switch, where you want to insert the values. In this example the values y1 and y2 are needed with 0 as well as with –1, the values u1, u2, u3 only with –1. Due to certain reasons (in order to be able to reconstruct the deferred values at any time) NN-Tool needs all values also with time step 0. So we are choosing at first all values with the time steps 0 and –1 and later on we set the values passive which are not needed. Determine the values correspondingly. Afterwards the window should look as follows.



With **Exit** the settings become valid and you return to the window Data Analysis.

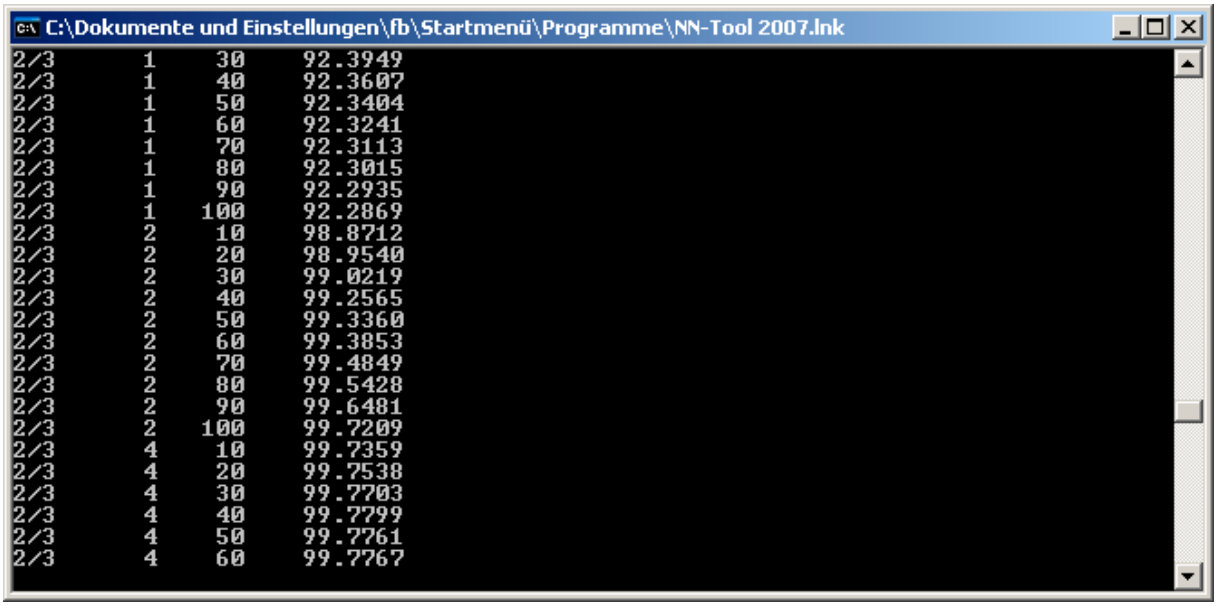


All parameters appear double now, at the present time value as well as a time step in the past (-1). Observe that the values at time step (-1) occur more rarely as they cannot be determined at the first record. Set the unneeded parameters u1, u2 and u3 passive at time step 0. Set the values y1 and y2 as output values. Now you should have 5 input values, namely all parameters at time step –1 as well as two output values.

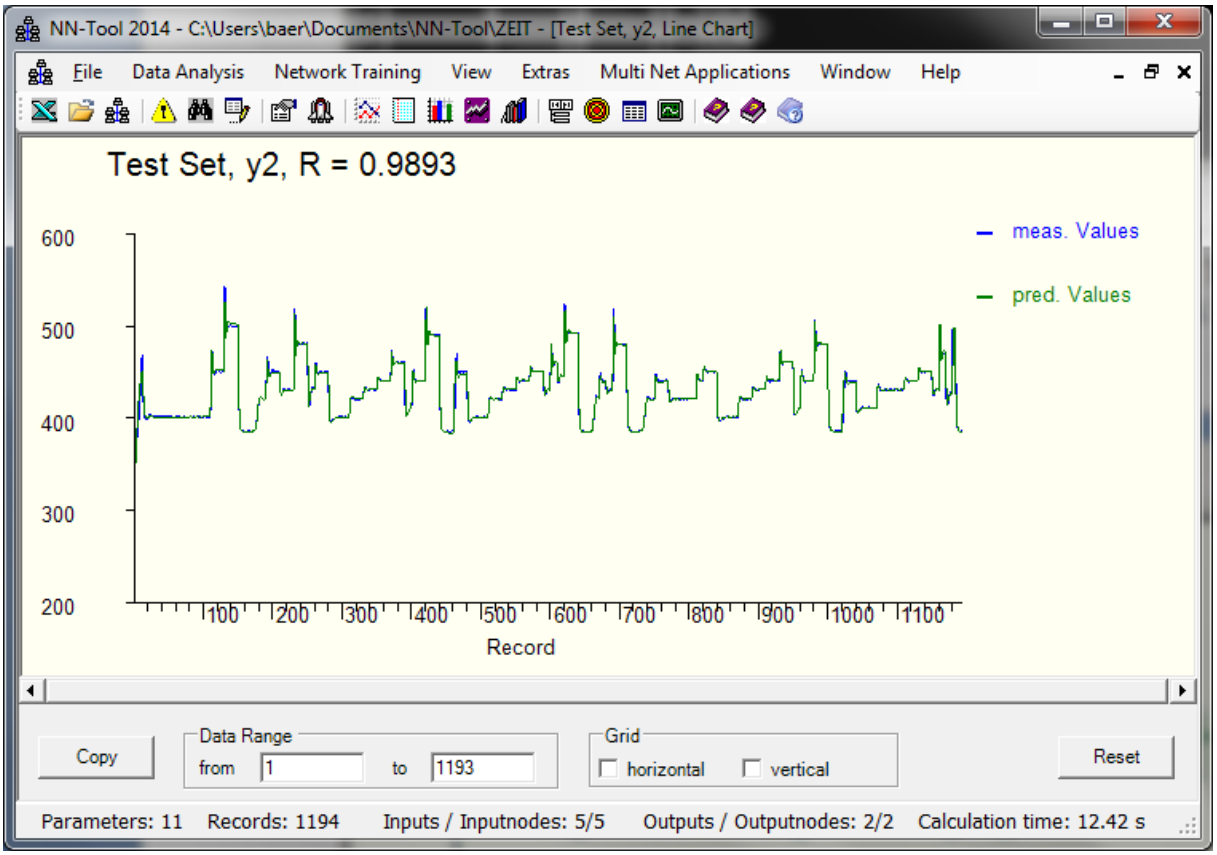


Now change to the tab page **"Training Parameters"**. Note that for the test set the option „Coherent Test Set“ has been chosen. NN-Tool has set this option as the default for time series. Alternatively, here comes the even stronger, but more numerical-expensive option **"n-**

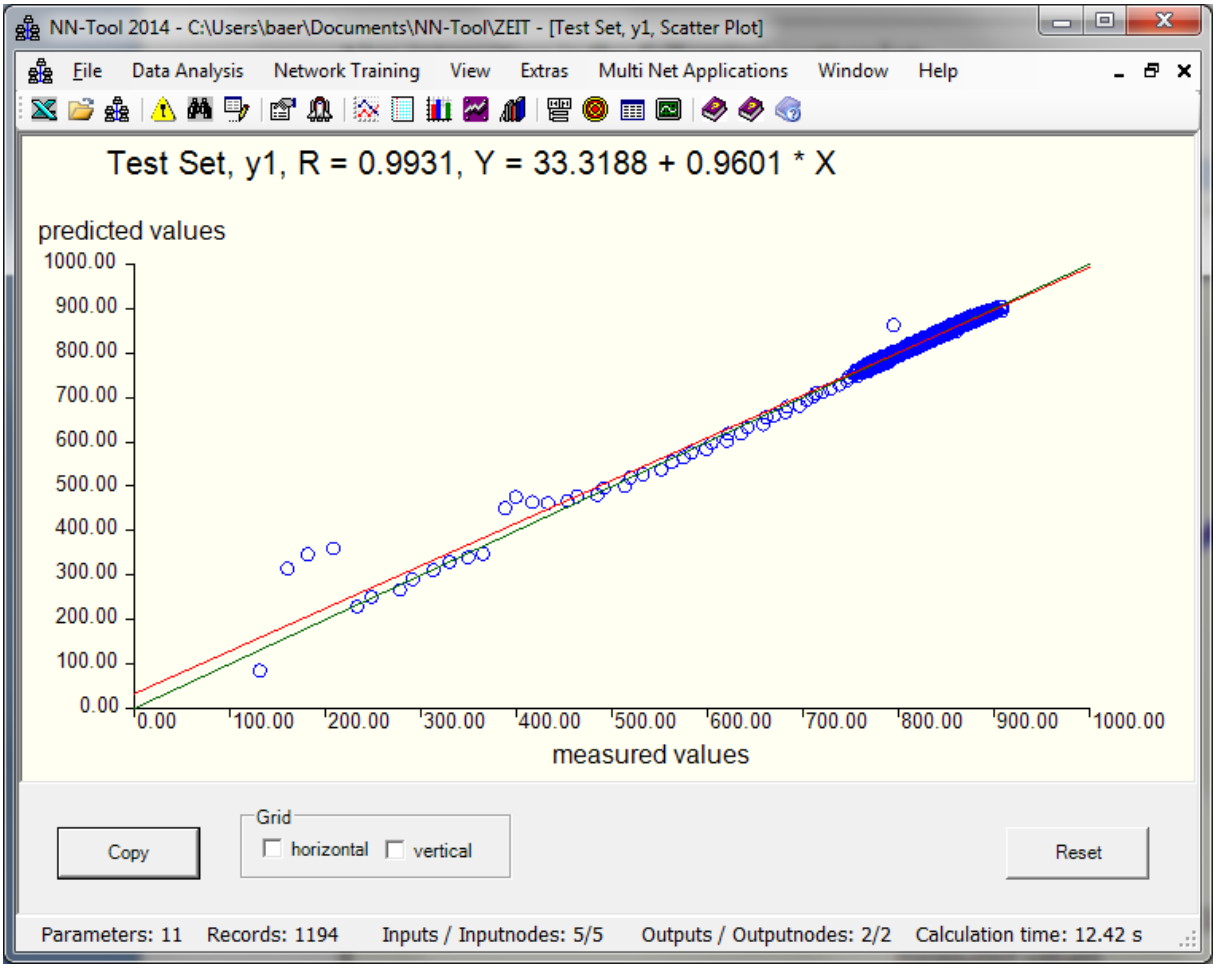
fold Crossvalidation in Blocks" into account. Choose this option and start the learning process. Note that now each output node will be calculated 5 times (for the 5 dynamically created test sets).



This is just the 2nd output with the third test set on a network with four internal nodes tested after 60 training steps. The correlation coefficient for the 3rd Testset is 99.7767%. At last determine the Line Diagram on the test set. The plot should look like:



Also interesting is the following scatterplot:



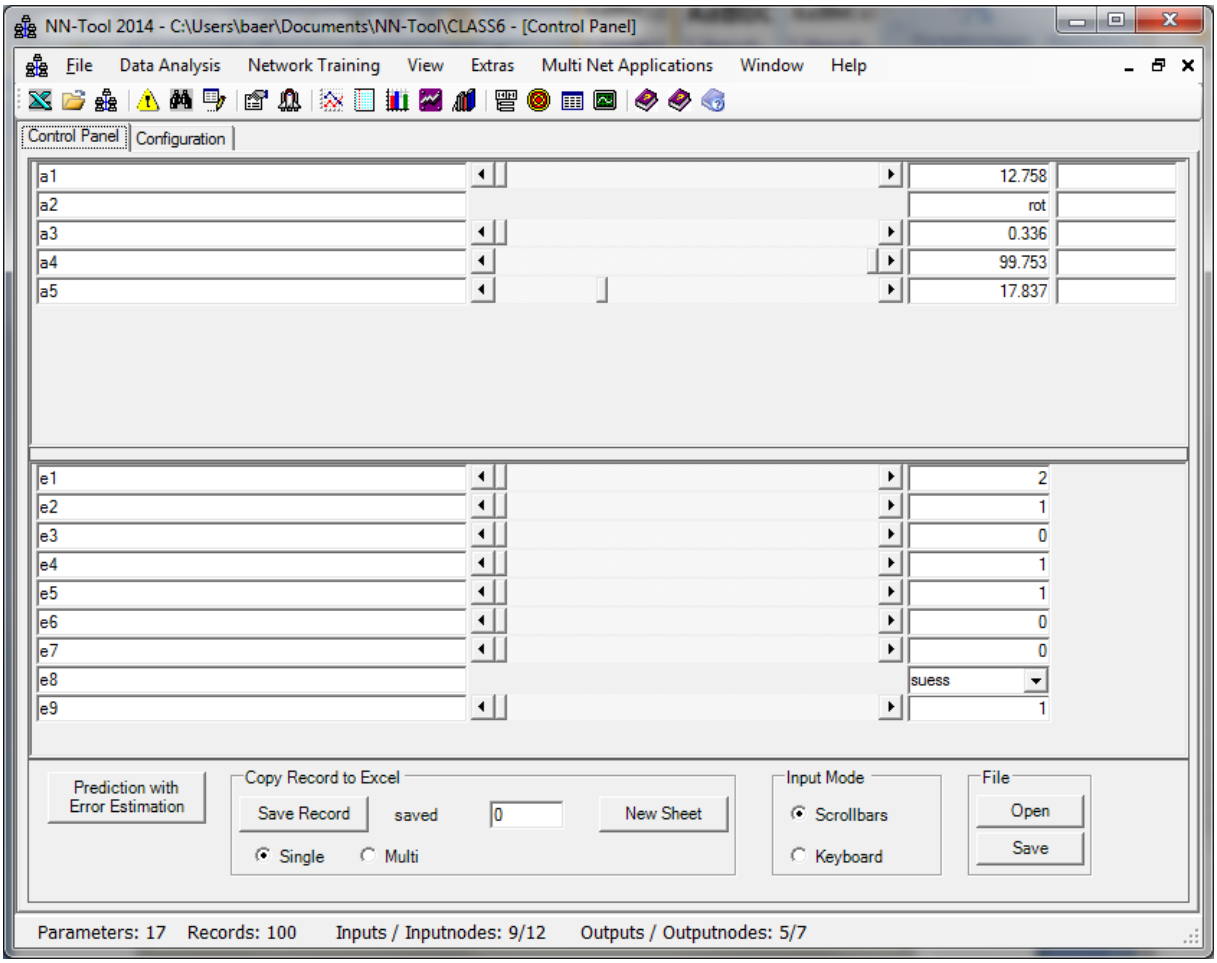
Question: what is the reason for the data oscillations at small values? Answer in the Annex "Dynamic Simulation".

Note: For larger data sets, we recommend network training with the "**n-fold Crossvalidation in Blocks**" option in "**Multithreading mode"**.

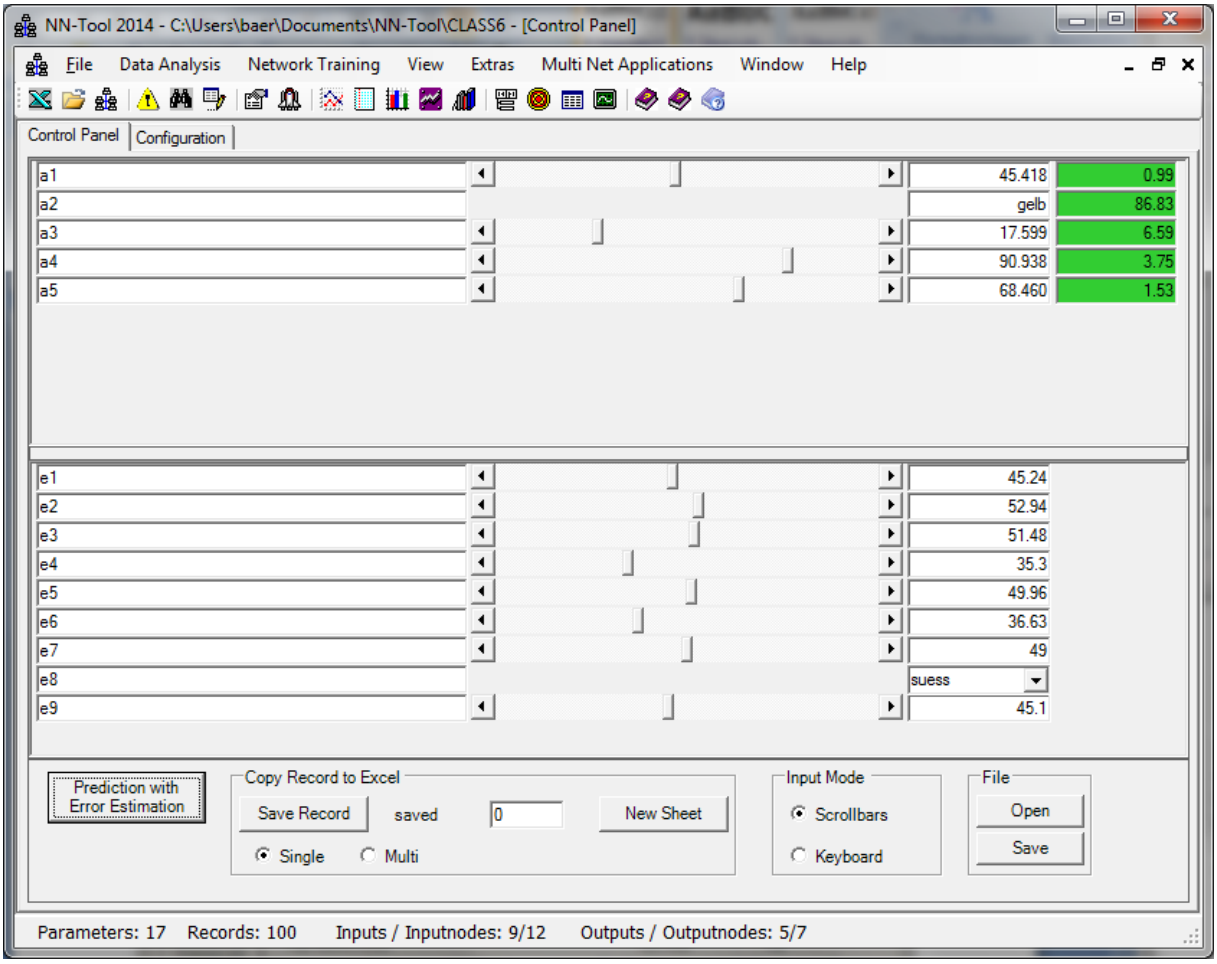
After this introduction to model creation with the NN tool, the two application modules "**Control Panel**" and "**Optimization**" are explained below. Further application modules can be found in the appendix.

9. Application Module Control Panel

The operation of NN-Tool application component Control Panel is explained by means of the example of application **class6** (see chapter 5). Start NN-Tool and load the created network class6. Now choose the menu point **Control Panel** in the menu **Extras.**

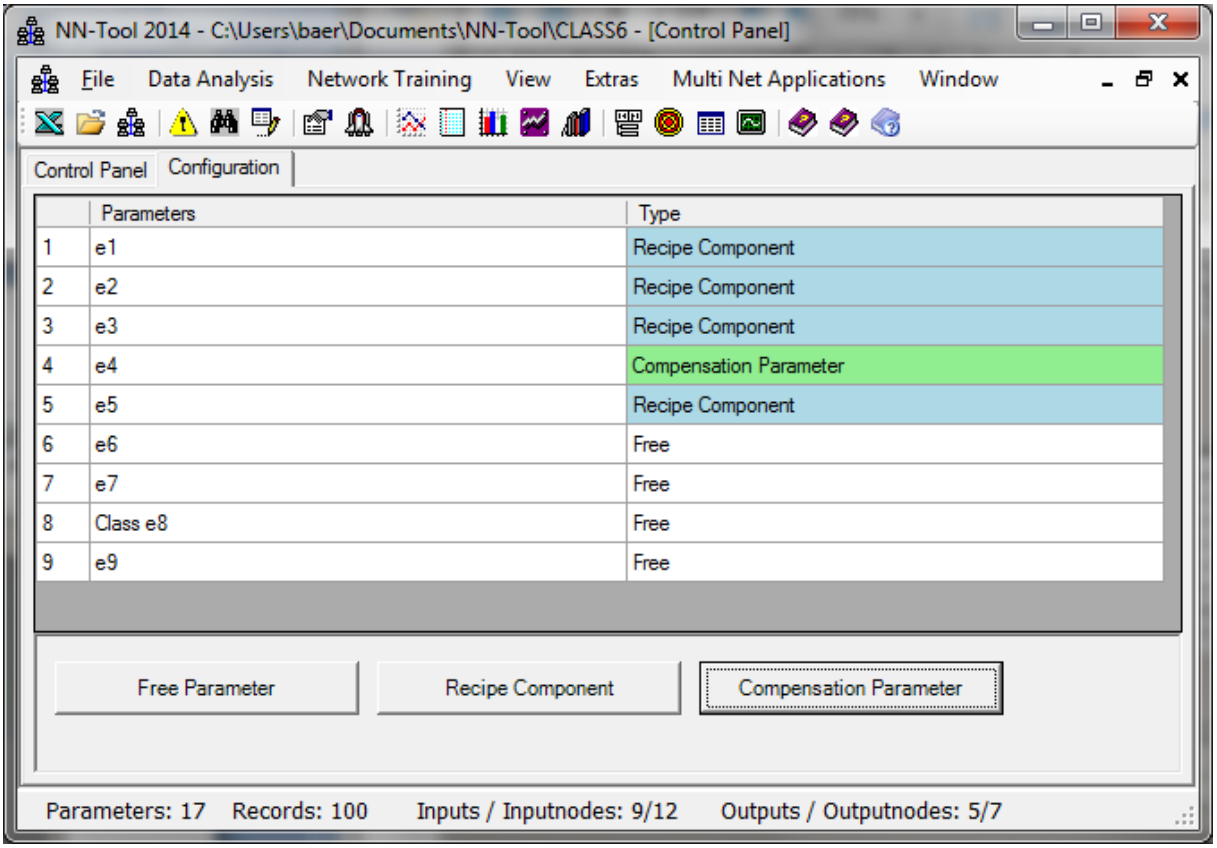


In this application predictions of new records can simply be made by setting the sliding controllers or in case of classifiers respectively over the option boxes. The numeric value of the controller setting is indicated in the corresponding field. Alternatively, inputs can be set on keyboard input mode. The settings serve to set the input values, the calculation of the corresponding output values is made automatically (also scrolling). Additionally error estimation can be demanded. Push the button "**Prediction with Error estimation"**. The following display appears:

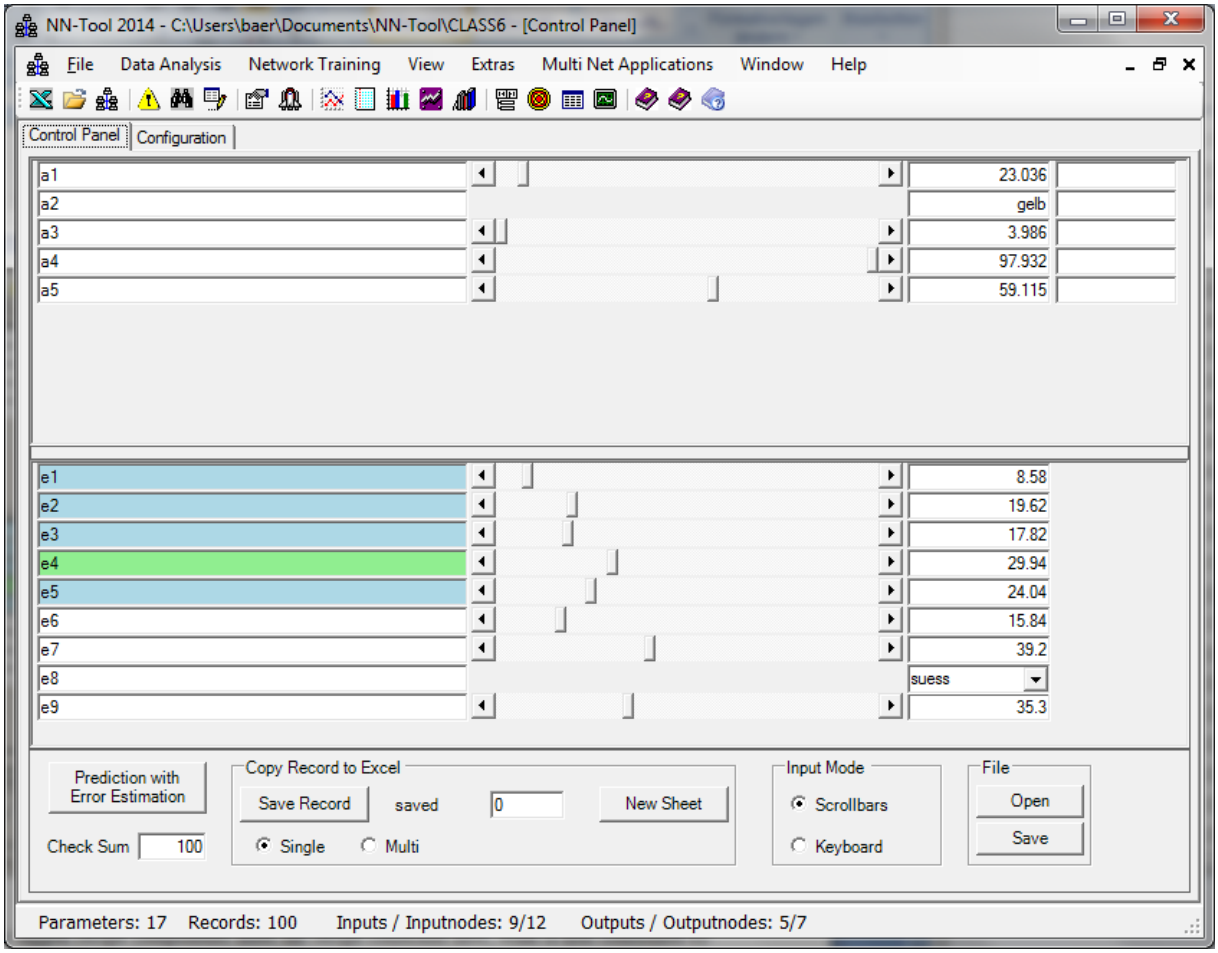


Estimated values are displayed for upper or lower deviations of an output parameter. **In case of classifiers the probability for correct classification is displayed (here 86.83 %).** For error estimation known records of the training set are taken, which are "close to" the record under consideration. The color indicates the validity of the estimation based on the number of known records in the vicinity of the actual record (dark red – bad, … , dark green – good).

If there are recipe components with the input values, these have to meet of course the Recipe Constraint: **Sum of concentrations of components = 100 %**. Instead of always adding up the values, the corresponding components can be defined over the tab page “Configuration”:



In the example the input values "e1" to "e5" have been defined as recipe components. The values "e6" to "e9" are furthermore freely changeable (e.g. temperatures and pressures). Moreover the component "e4" has been determined as "Compensation Parameter". That means that the sliding controller for "e4" cannot be adjusted manually anymore, but is always readjusted that way that the recipe side condition is met (if possible).
After switching to the tab "Control Panel" and using the sliders the following picture results:



The color tagged recipe components meet the recipe constraint now, what is also confirmed by the field “Check Sum”. By means of the buttons "Save" and "Open" the configurations can be saved or loaded respectively in .mix-files. By the button "Save Record” the results can be transferred to Excel.

10. Complex Optimization Problems – Recipe Optimization

10a) Application Range of the Optimizer

Under the menu Extras, NN-Tool makes available an environment for definition and solution of complex optimization problems. The optimizer makes it possible to calculate optimum recipes (mixtures) and operation points on the basis of a neural network created with NN-Tool. With the optimization a lot of side conditions can be considered. For the input values applies in detail:

- For each input value it can be determined whether it is also optimized or kept on its initial value. With the optimization upper and lower limits can be fixed.
- The input values can be allocated to any component groups (e.g. group of catalysts, stabilizers, flame-retardants etc.). For each of those groups upper and lower limits as well as the maximum number of used components can be stipulated. Example: from the 20 available catalysts not more than 3 are to be used with a total concentration from 2 to 5 percent. At the same time exactly one flame-retardant with a concentration of 5 % is to be used.
- Beside the recipe components further continuos values, e.g. pressure, temperature, residence time, can also be considered, i.e. these values can either also be optimized or kept on their default values.
- As further input values, additional classifiers can also be optimized. For instance the optimum recipe together with the most suitable mixertype can be determined.

For the output values the following options are available:

- Detailed target values can be aimed or it is only prevented that certain quality features are exceeded or fall short of. Moreover permissible band width can be stipulated.
- Classifiers can also be optimized: "calculate color neutral mixture with the following further characteristics..."
- Recipe costs can also be optimized.

Three solution strategies are available, that offers different "creativity" to the system:

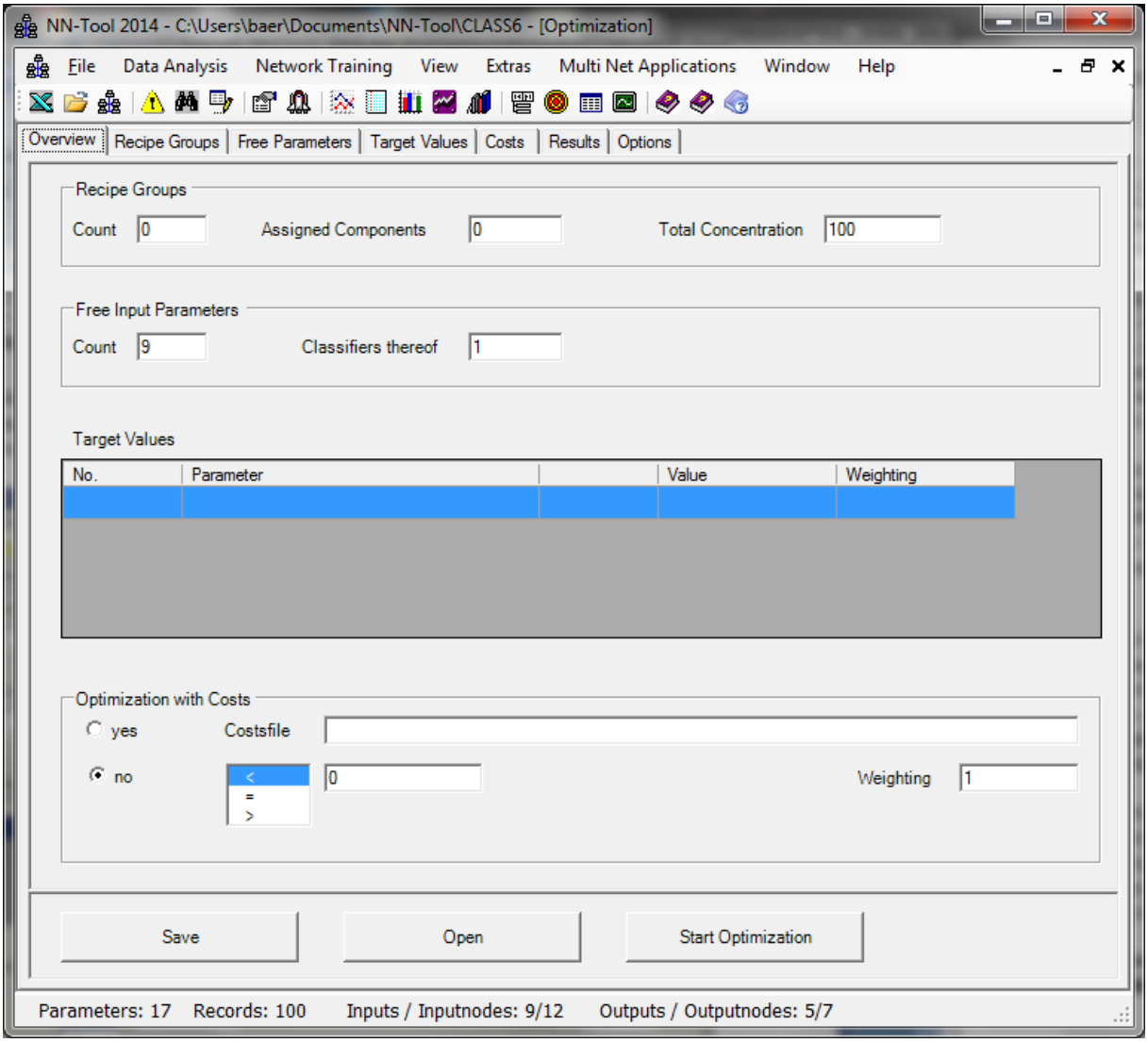
- The system creates the recipe to be calculated entirely from start-up (maximum creativity).
- The system proceeds on existing recipes, but it can replace used components by others.
- The system determines the optimum existing recipe and varies only the concentrations, i.e. especially it does not replace any components (minimum creativity).

The consideration of recipe components is completely unnecessary. In this case the system provides a calculation of optimum operation points.

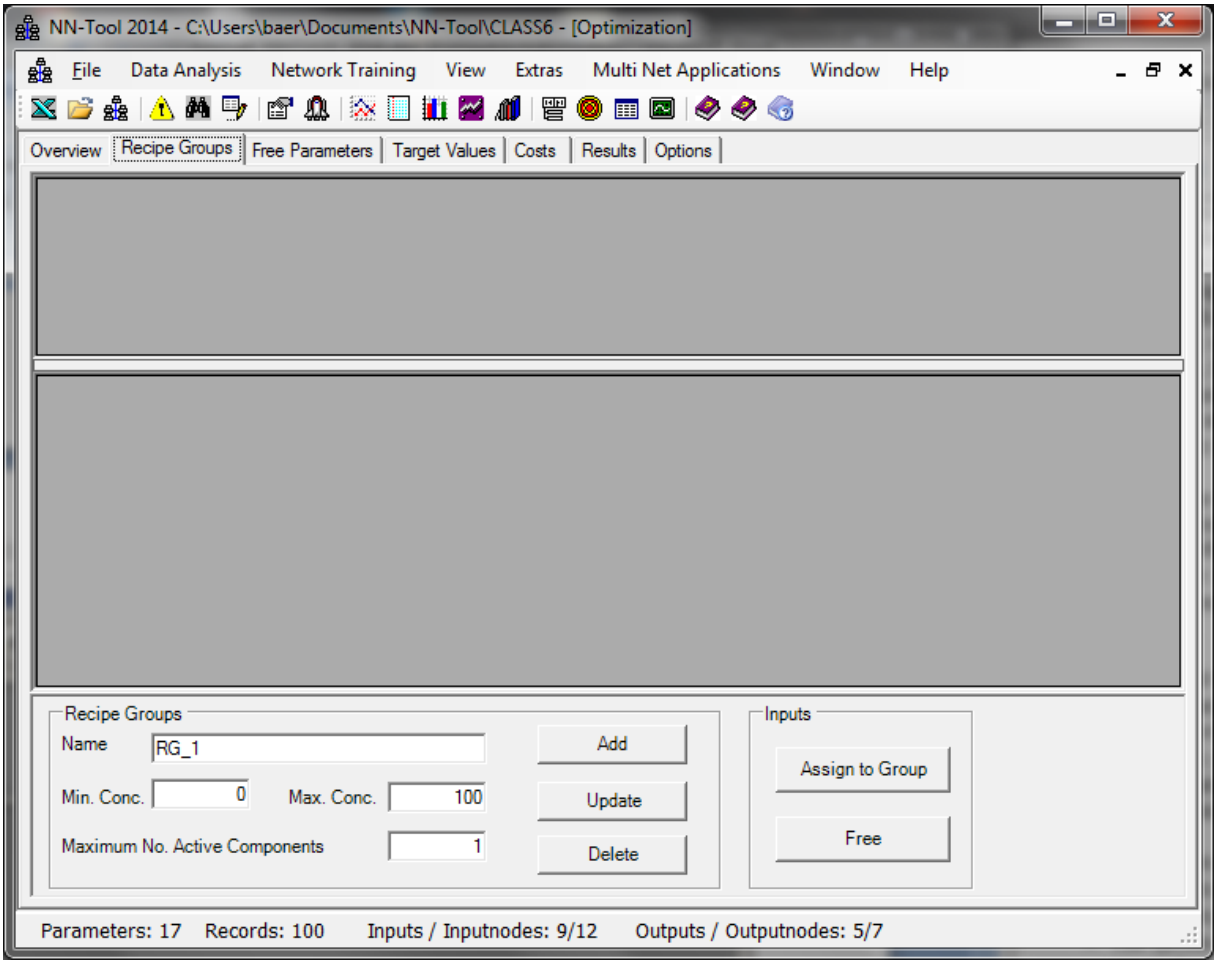
[1] The real optimizer can run independently and therefore can also be used outside NN-Tool. (after corresponding license has been granted). In the first place it is conceived for integrating in end user applications (especially databases) and does not have a special user surface. Because the program is controlled over input files and is writing the results in an output file, integration in any environments (e.g. Oracle databases) should be possible without problems.

10b) Definition of Optimization Problems

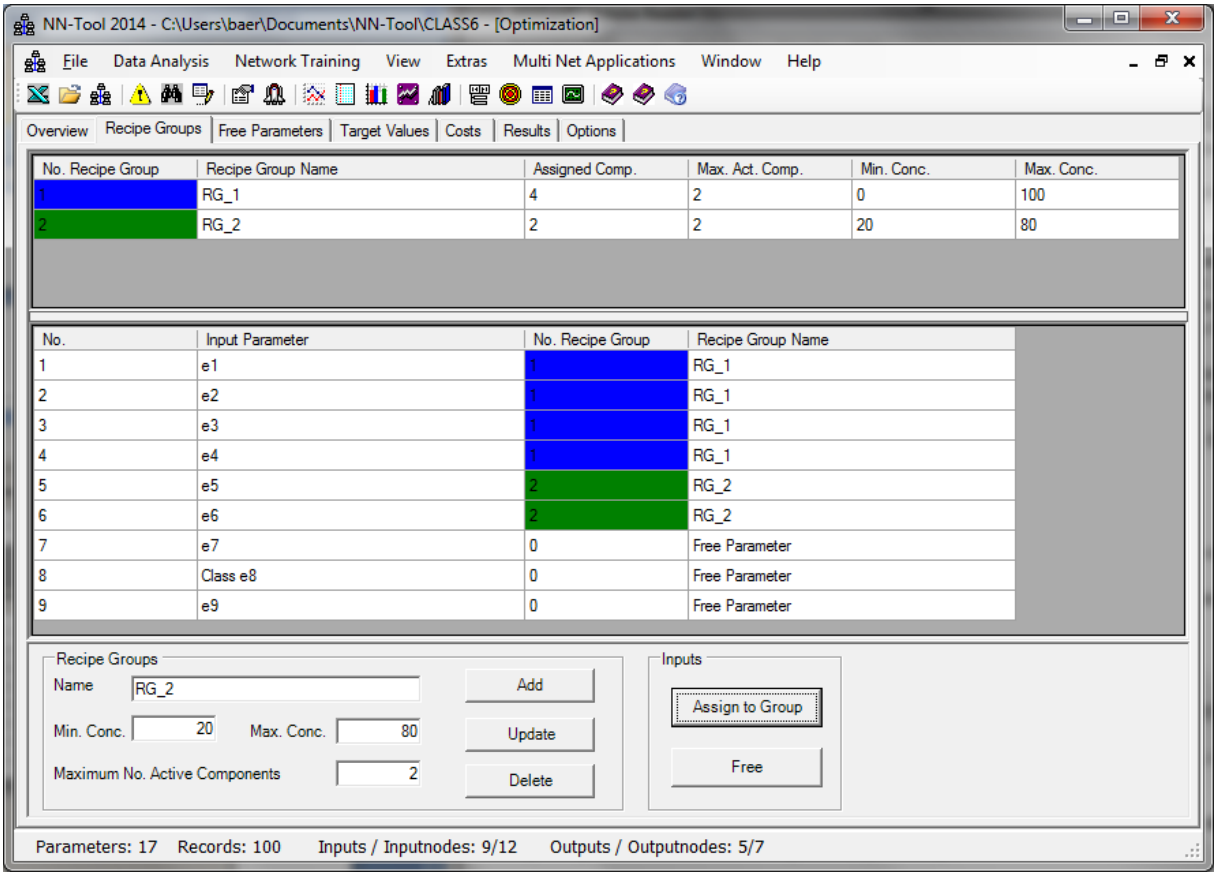
The procedure for definition and solution of optimization problems is executed by means of a recipe optimization with the application **class6** (see chapter 5). Start NN-Tool and load the created network **class6**. Choose the menu point **Optimization** in the menu **Extras**.



At first we have to define the recipe groups, which have to be considered. Change to tab page **Recipe Groups.**



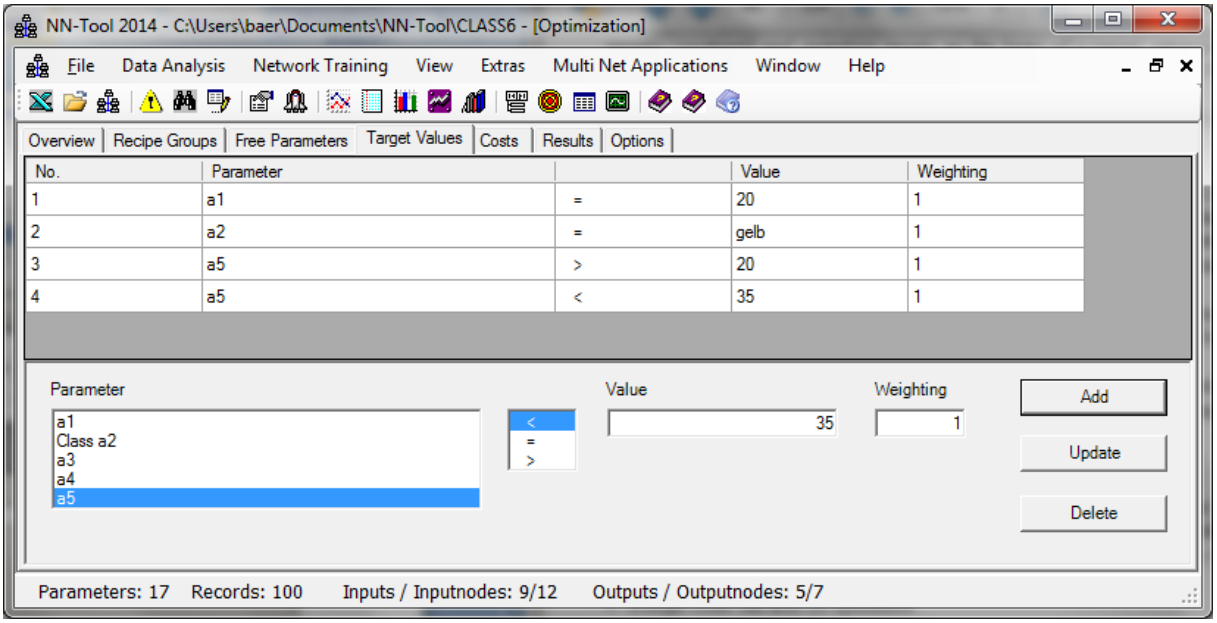
Define two recipe groups (RG_1 and RG_2). The parameters e1, e2, e3 and e4 have to be assigned to group "RG_1". The parameters e5 and e6 belong to group "RG_2" (at first define the groups, and then assign the components). The group "RG_1" is to be allowed to vary over the complete range from 0 to 100 per cent, but only have 2 active parameters (i.e. components) maximally. This means for the group "RG_1" that maximally 2 of the 4 assigned components are to be allowed to come up in the calculated recipe. The group "RG_2" is to be allowed to vary from 20 to 80 % with 2 active parameters. Finally the window should look as follows:



Go to the tab “Overview”. The tab indicates that 2 recipe groups have been defined and 6 components belong to them. Furthermore there are 3 other parameters (e7, e9 and classifier e8). With the tab “Free Parameters” you can define limits for the other parameters. The variation range of the parameters is used for default settings of limits. Adopt these settings. Now specify the following settings for the output parameters by using the tab page **“Target Values”**:

a1 = 20
a2 = „gelb“
a5 >= 20
a5 <= 35

Afterwards the tab page “Target Values” should look as follows:

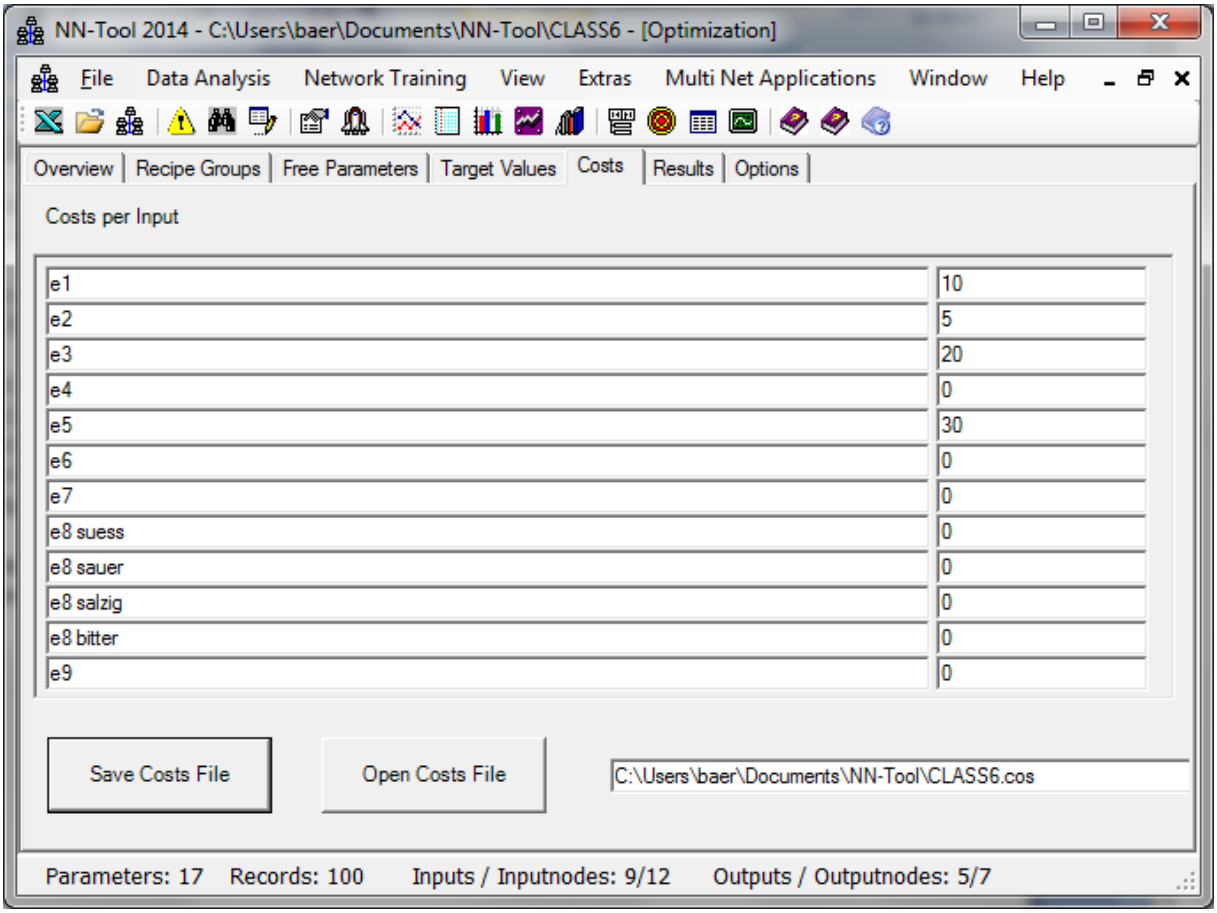


By setting the weighting (positive number) the different meaning of the single optimization targets can be taken into account.

With the recipe optimization the costs should be considered and therefore a costs file has to be created. Change to tab page “Costs” and define the following costs:

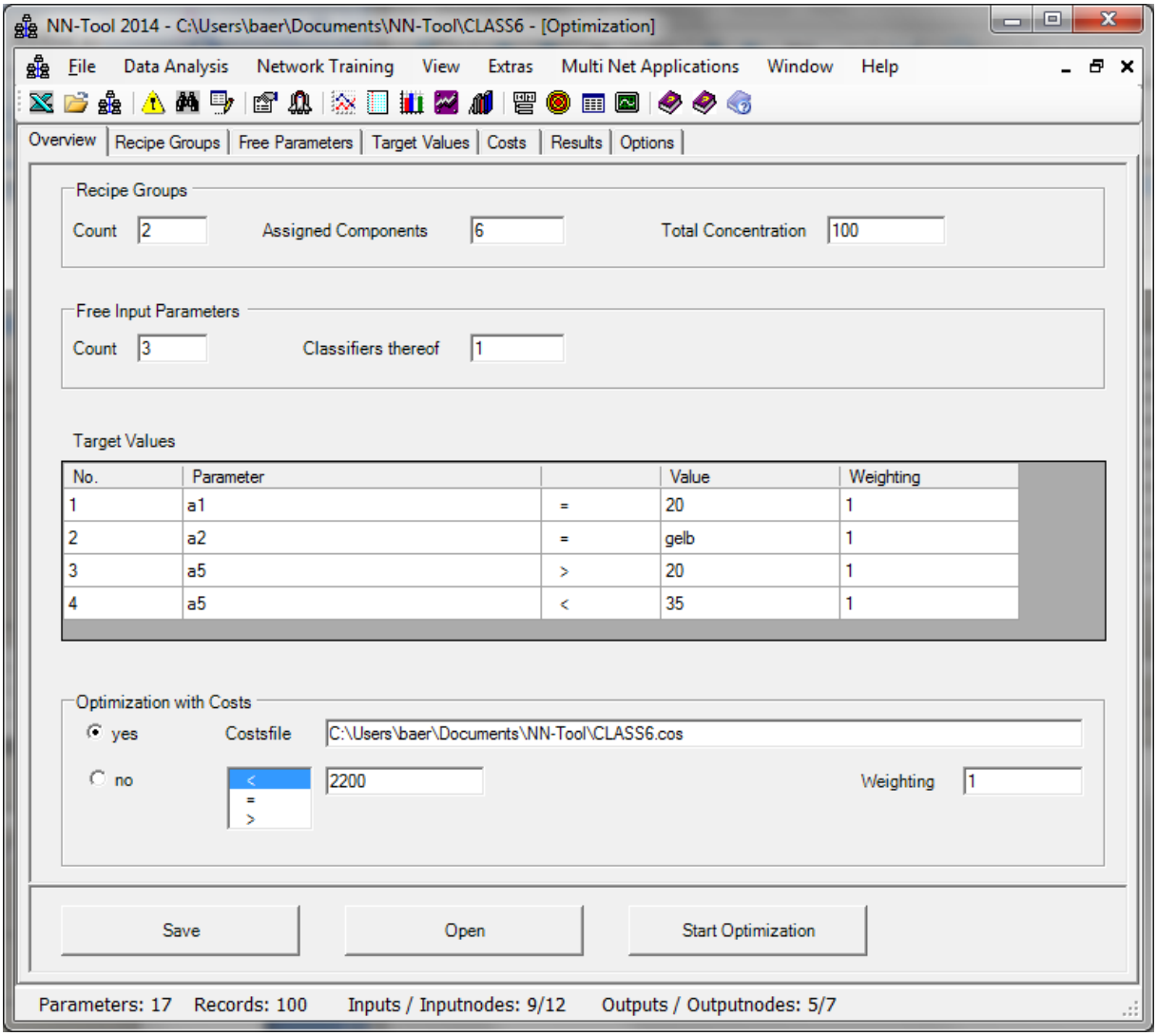
e1 = 10
e2 = 5
e3 = 20
e5 = 30
Rest 0

That means that each % of the component e1 is assessed with 10 cost units, etc.
Save these costs in the file class6.cos.



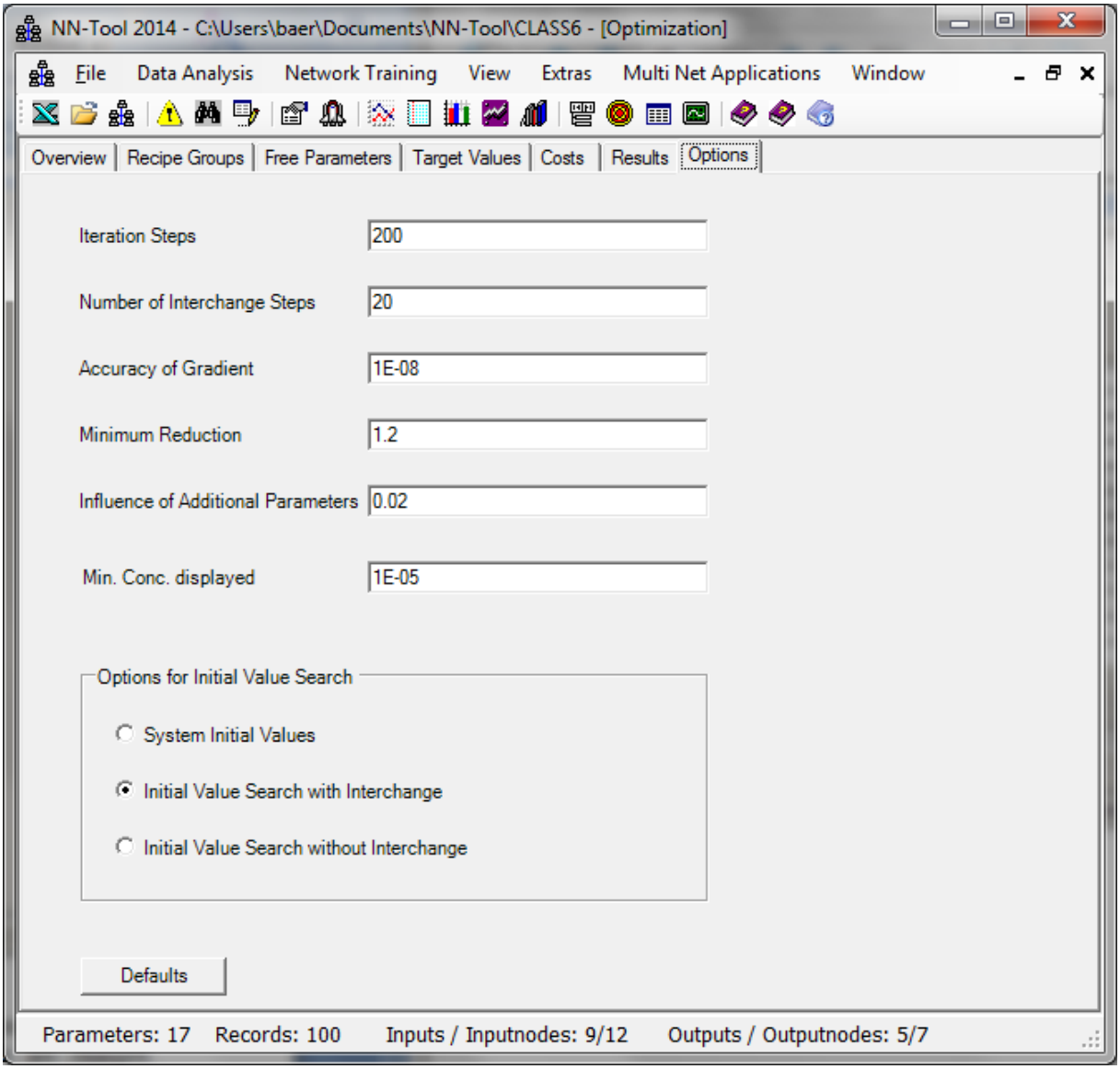
Now change to “Overview”.
Set as cost target value 2200 with the setting "<" and weighting 1.

The main window should look as follows now:



With **Options** tab the numerical parameters of the optimization process can be influenced. Principally the calculation method consists of three phases.

1. Determination of an initial recipe, fulfilling all side conditions. With that you can start on the basis of System Initial Values (1st Option Initial Value Search) or search for suitable start recipes in the .ptp-file (Options Initial Value Search 2 or 3).
2. Optimization of recipe concentrations by means of an adjusted numerical procedure (asymptotic CG-procedure, projected on the side conditions). With that so-called iteration steps (gradient steps) are executed in order to minimize the so-called target functions. The lower the value of the target functions, the more exact the targets will be achieved. Typical number approx. 200.
3. Interchangesteps: The components in the recipe are replaced by more suitable components. This step is not executed with the Option "Initial Value Search without Interchange".
4. The steps 2nd and 3rd are executed alternately (except with Initial Value Option 3).



Iteration Steps give the maximum number of gradient steps between the interchange steps. During the gradient steps only the concentrations of used components are changed. Recommended range: 50-500
(higher value – longer calculation time, better results).

Number of Interchange Steps, i.e. number of operations with changes of used components.
Recommended range: 5-50
(higher value – longer calculation time, better results).

Accuracy of Gradient. The gradient is calculated by numerical differences. That is the more exact, the smaller the value is. Too small values, however, lead to numerical instabilities.
Recommended range: 1.0e-8

Minimum Reduction. Demanded reduction factor of target function by gradient steps. The algorithm stops, when after an interchange step no more sufficient reduction of the target function could be reached by the following gradient steps. A value of 1.2 means that the target value has to be reduced at least by a factor of 1.2, otherwise the algorithm has to stop. The value has to be higher than 1.

Recommended range: 1.01 – 2.0
(higher value – shorter calculation time, weaker results).

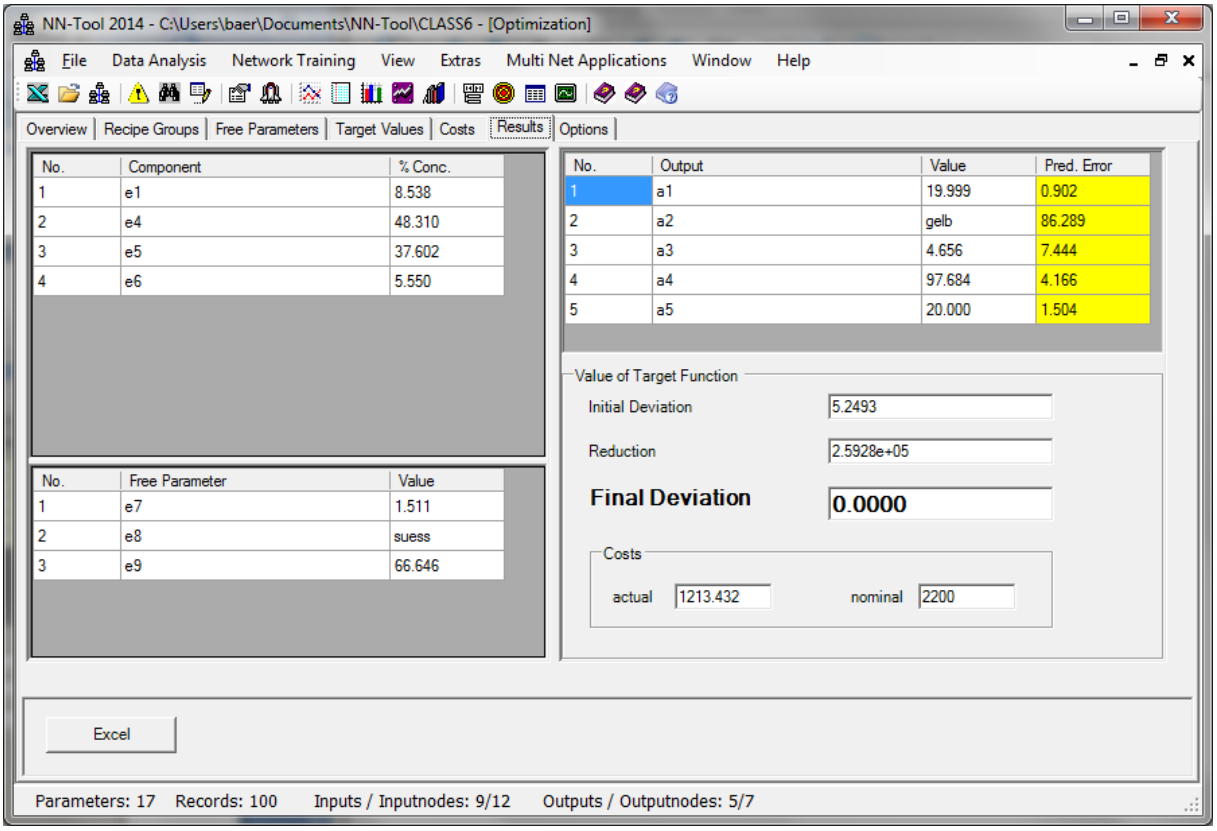
Influence of Additional Parameters. Weighting of additional continuous variables. If beside pure recipe components, which can only assume values from 0 to 100 (indication in %) as concentrations, additional continuous variables are optimized, these have to be transformed appropriately (i.e. scaled in their amount) for coming in the same order like the components. The maximum possible values of the transformed variable correspond to the reciprocal of the indicated parameter. A parameter of 0.02 means that the transformed variable, regardless of its original physical value, can assume only values from 0 to 50 **(**$= 1/0.02$**),** or from – 50 to 50 for variables with different signs, during the optimization. In the result, of course, the variable is re-transformed.

Recommended range: 0.01 – 0.2
(no clear influence on the calculation time)

Options for Initial Value Search. There are three possibilities for determination of this control value:

1. The algorithm uses the start vector, which is included in the file nnoptin.dat. The components can be replaced.
2. The algorithm looks for a suitable initial recipe in the file <application>.ptp, which fits as good as possible to the side conditions and the target values. During the optimization the components can be replaced.
3. The algorithm looks for a suitable initial recipe in the file <application>.ptp, which fits as good as possible to the side conditions and the target values. During the optimization the components are kept. With that the system can only vary already known recipes, but cannot insert new components.

Adopt the default settings. Return to the main window "Overview" and save all settings in the file **class6.opti**. Now push the button **Start Optimization**. The optimization runs as an independent application (C-Program), also independent of NN-Tool. After finalizing the optimization the following window is shown:



The calculated values of the used recipe components are on the upper left side (only these parameters fulfill sum = 100), and underneath are the values for the other parameters. In the shown example the default values for the output parameters as well as the costs could almost be achieved. That is shown by the value 0,0000 for the remaining final deviation of the target function. The reduction value indicates how much the original value of the target function has been reduced. By means of the button **Excel** the calculated solution can be transferred to Excel.

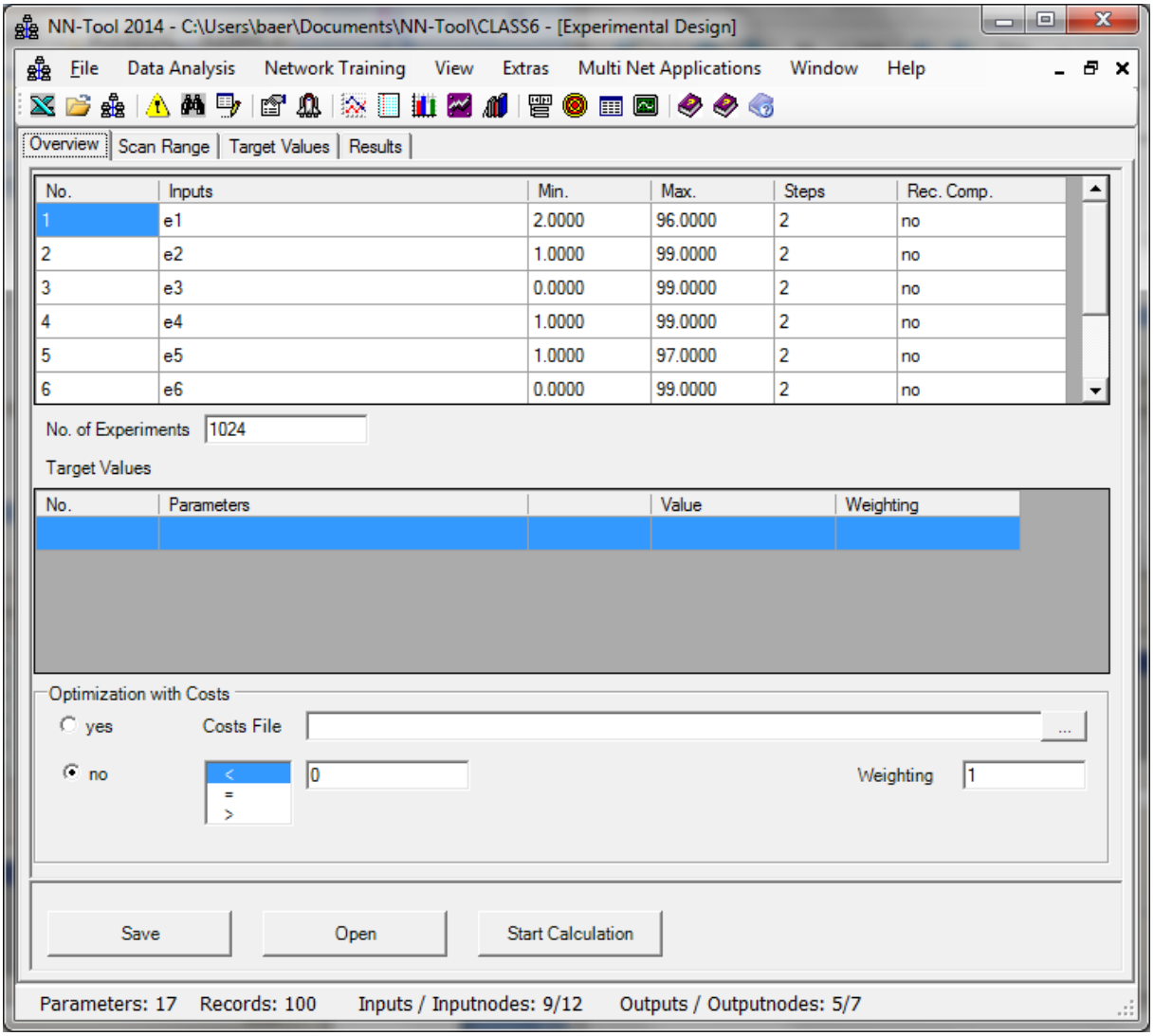
Finally: Please inform me about occurred errors as soon as possible, also in case that these are "only" operating errors.

Email: info@baermann.de
Tel.: ++49 211 7489973 (answering machine)

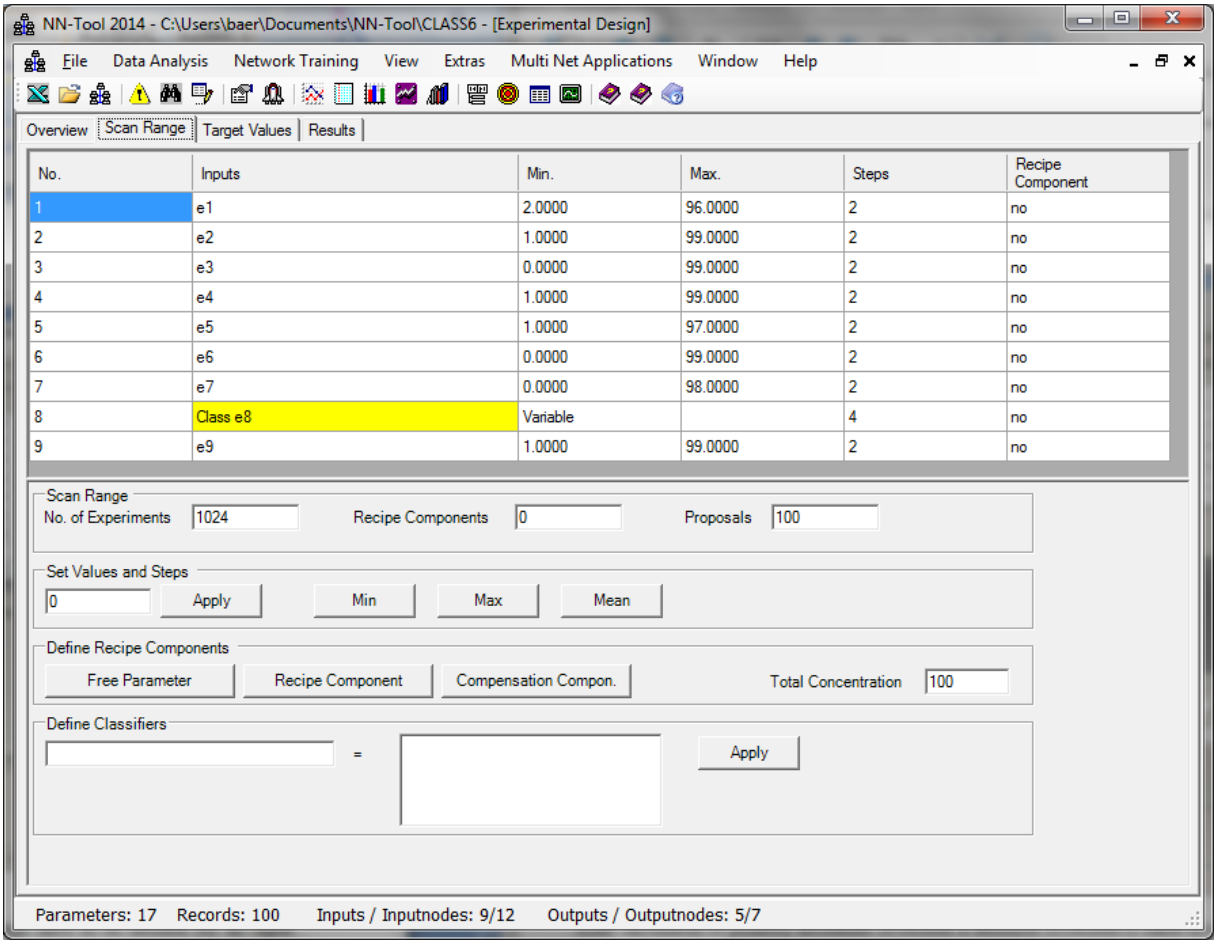
Only this way the "last but one bug" can be found.

Annex 1: Experimental Design

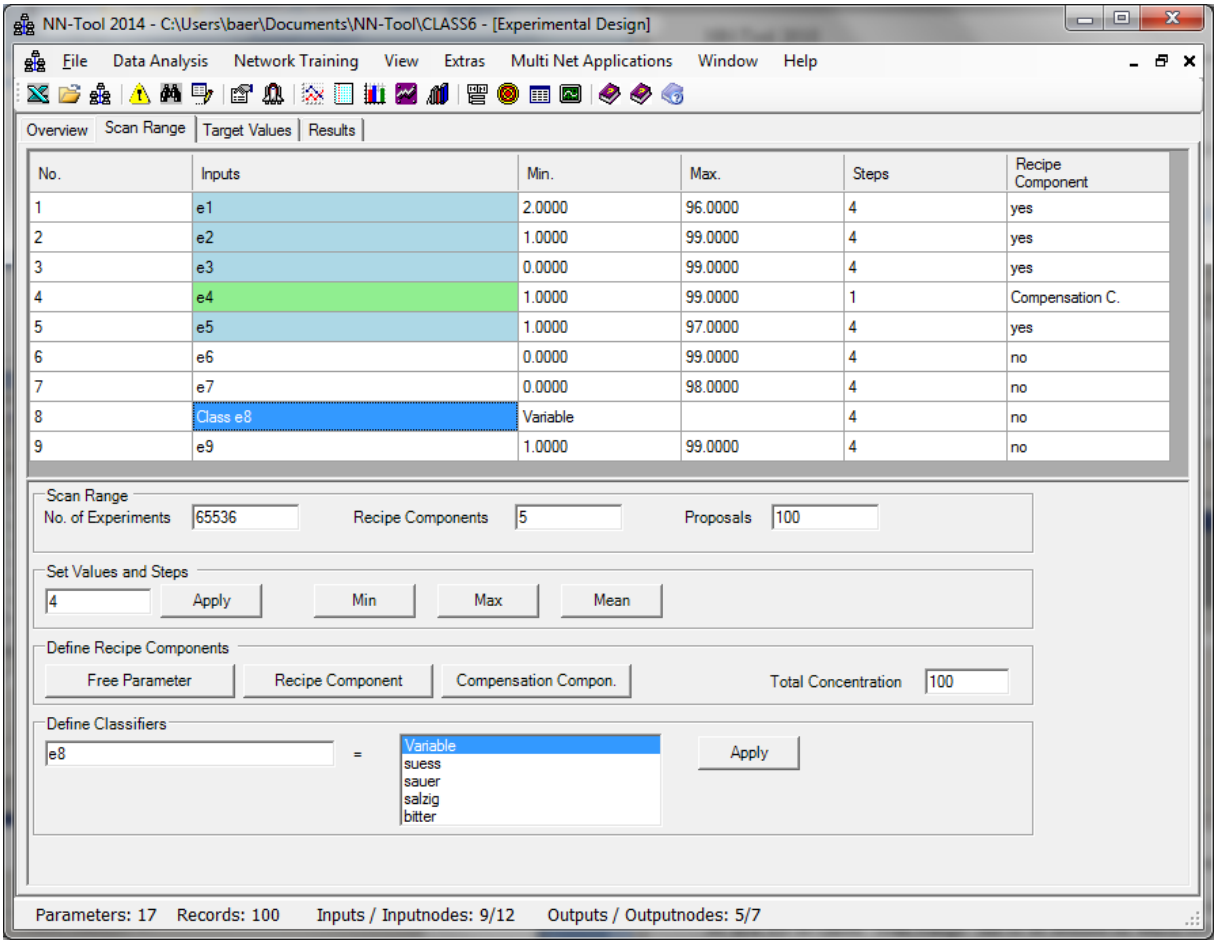
The module **Experimental Design** (menu Extras) offers the prediction of a large number of automatically calculated records.



After calculation of possibly thousands of records a selection of records is made according to the chosen target values.
A user-specified number of records regarding the most appropriate target values is displayed.
At first the so-called “Scan Range” has to be defined on which the analysis is to be made. Go to “Scan Range”:



Here a minimum and a maximum value of the scan range have to be defined for all input parameters. Furthermore the number of steps (levels) between them has to be defined. In a setting such as for example 30, 60 and 3 the corresponding parameter is tested with 30, 45 and 60. Finally all combinations are predicted and the best regarding the target values are proposed. The number of proposals can be adjusted in the corresponding field. Classifiers can be tested on all classes ("Variable") or a class can be given. Moreover parameters can be defined as recipe components. In this case they must add up to total concentration (usually 100). This is realized by a compensation component (this should have the highest possible range). An adequate configuration could look like this:



Here are all the inputs will be evaluated on 4 levels with the exception of the compensation component, which always needs to be adjusted so that the recipe condition "sum equal to 100% is met. A total of 65,536 variants will be calculated and the best 100 regarding targets and costs will be proposed. The definition of costs and targets is analogous to the complex optimization module. A required costs file has to be defined using the optimization module.

Note: Observe that with the definition of recipe components there is the possibility to specify incompatible constraints. In this case, of course, a suitable dataset cannot be found.

Annex 2: Automatic Documentation

This feature in the menu "**Network Training**" makes it possible to generate fully automatic all relevant information for a network and to save it in an Excel-file. This also includes graphics like scatter- and line plots.

Annex 3: Batch File

Besides the usual graphical interactive operation, NN-Tool can so to speak be operated by remote control over a so-called batch file (corresponds to a macro recorder). This is especially interesting, when always the same network structure with always different datasets is to be trained. Another possibility is the integration of NN-Tool in user specific applications.
A corresponding file has the extension .cob and can be created automatically after the training process. By double-clicking on a .cob file, NN-Tool then carries out the entire modeling fully automatically.
The feature can be found in the main menu "**Extras**", menu item "**Create Batchfile**"

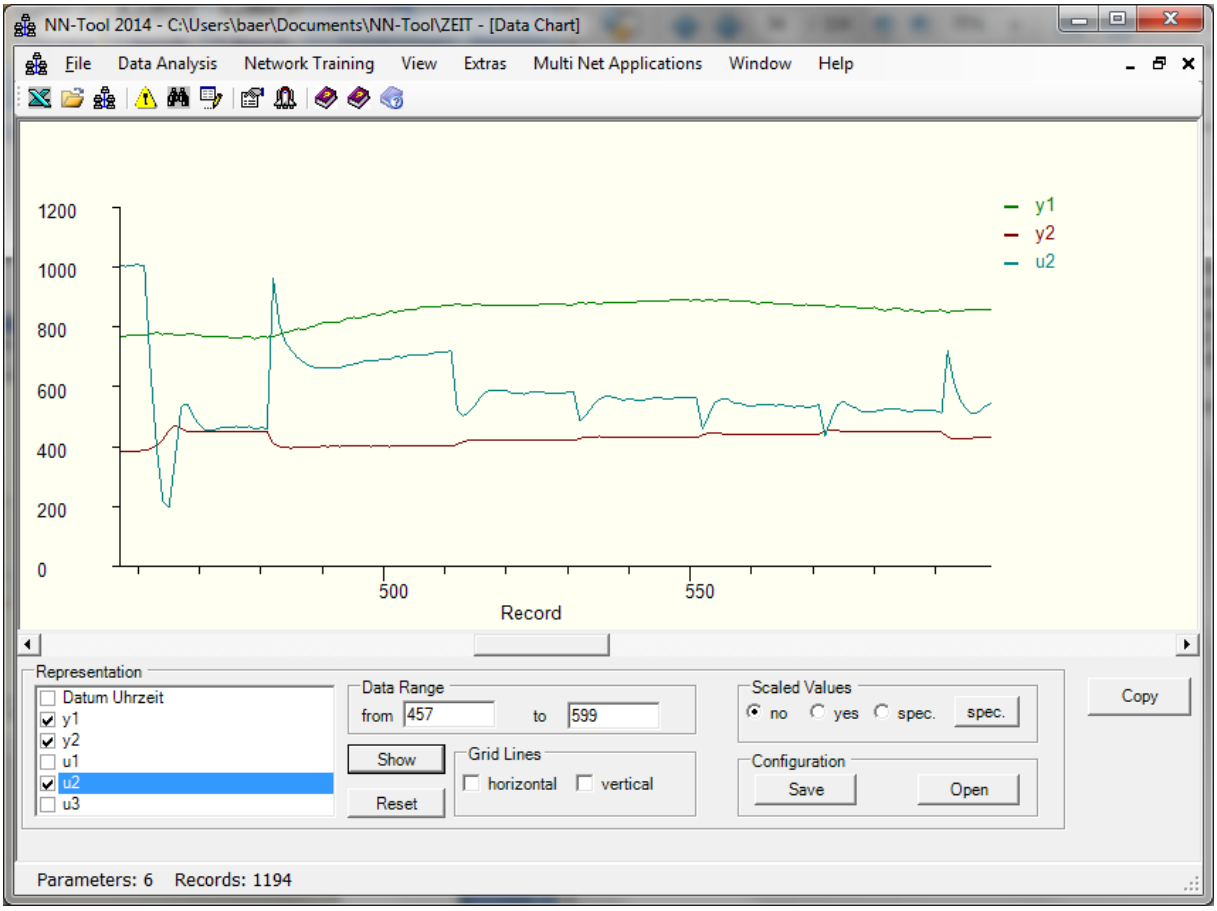
Note: This feature currently supports only static networks (i.e. no time series). However, these can be shifted externally.

Annex 4: Special Functions for Data Analysis and Preprocessing

In this chapter the special functions in the menu “Data Analysis” and the function “Histogram” (start from window “Data Analysis”) are described. The functions “Data Chart”, "Histogram", "Cross-Correlations" and "Optimum Inputs" serve for the more detailed analysis of single parameters, for the automatic delay time estimation as well as for the determination of optimum input parameters with networks having a lot of parameters.

A. Data Chart

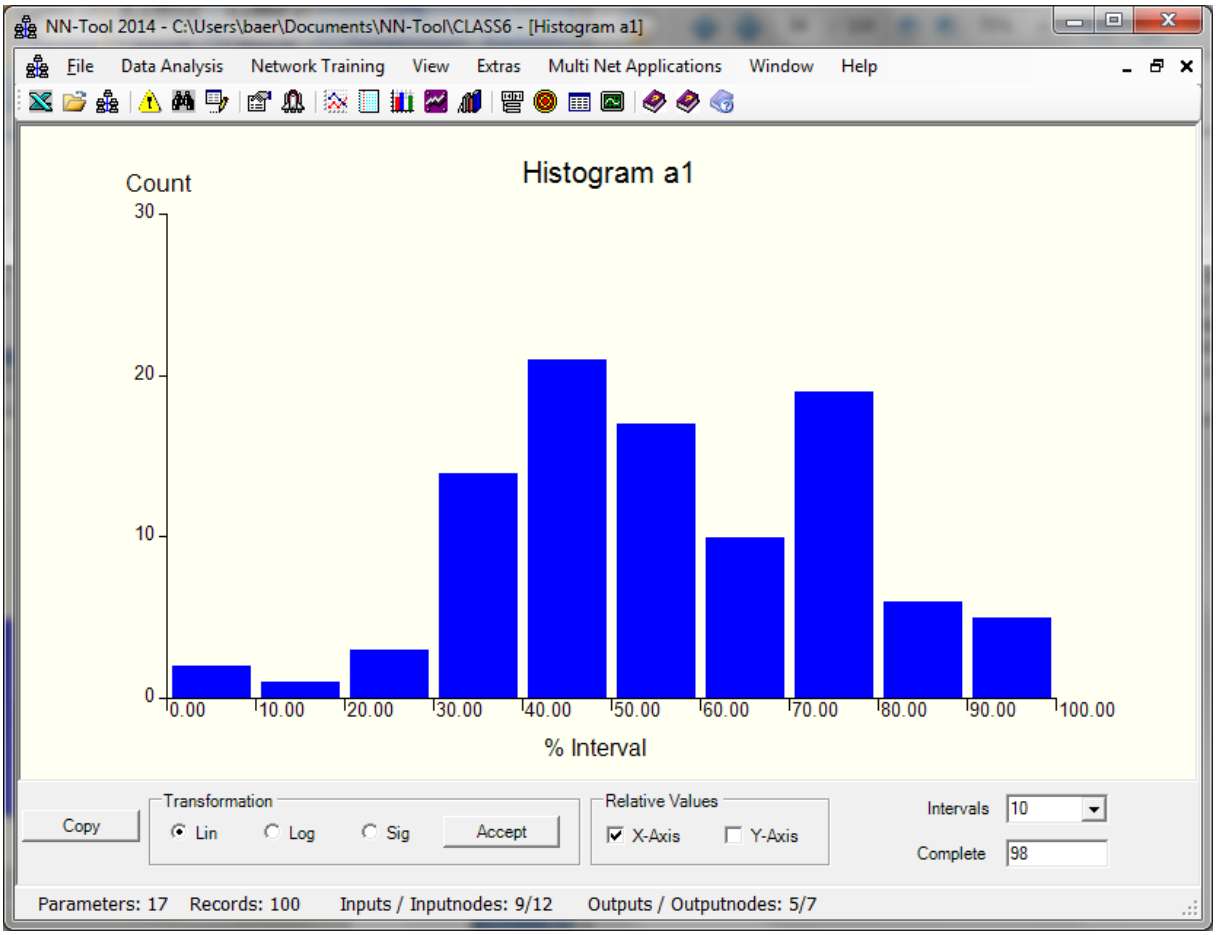
With the window **„Data Chart“** several selected parameters can be displayed simultaneously depending on the record number.



If different parameters have different magnitude, we recommend that the option "Scaled Values" or a specific scale (option "spec") is used. Using the left mouse button allows to select individual graphic points for detailed information. By pressing and pulling with the right mouse button the graphic can be zoomed. The middle mouse button resets the zoom option and all existing records are displayed.

B. Histogram

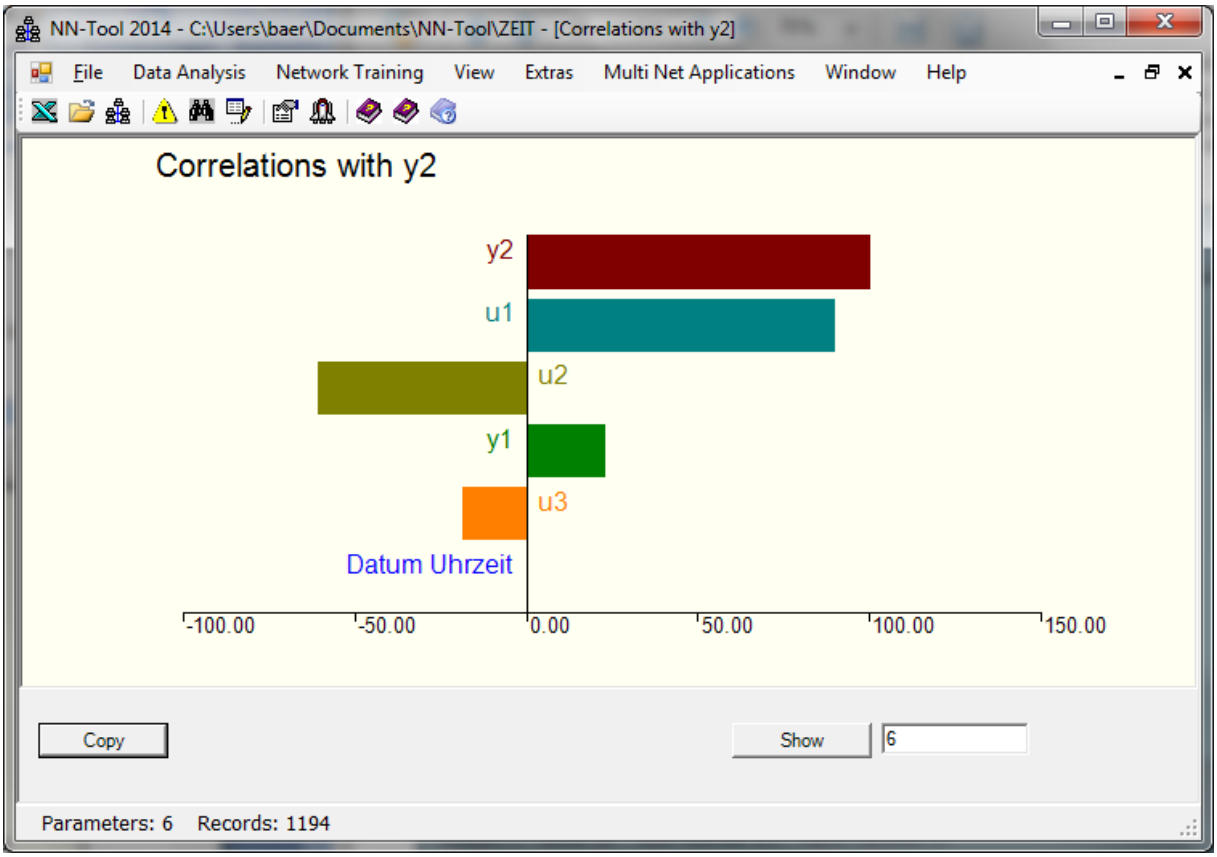
The form **"Histogram"** describes the frequency distribution (histogram) of selected parameters (Start from window "Data Analysis" by double-click on a parameter name). The number or percentage of records, for which a certain parameter is assigned to a certain partial interval, is shown above the position of the interval. The number of partial intervals can be set.



Main application area: Ideally parameters should be distributed as even as possible over their range, i.e. the interval from the minimum to the maximum value. I.e. in essence the histogram should be constant on the optimum value (100% divided by the number of intervals), which defines the so-called uniform distribution (not realized here). That is especially important for a good training of output parameters. If single parameters (especially output parameters) differ strongly from the uniform distribution, a suitable transformation (e.g. transition to the logarithm of this parameter) should be taken into consideration.

C. Cross-Correlations

A parameter is selected in the lower area and this form displays the correlations of all other parameters to it.

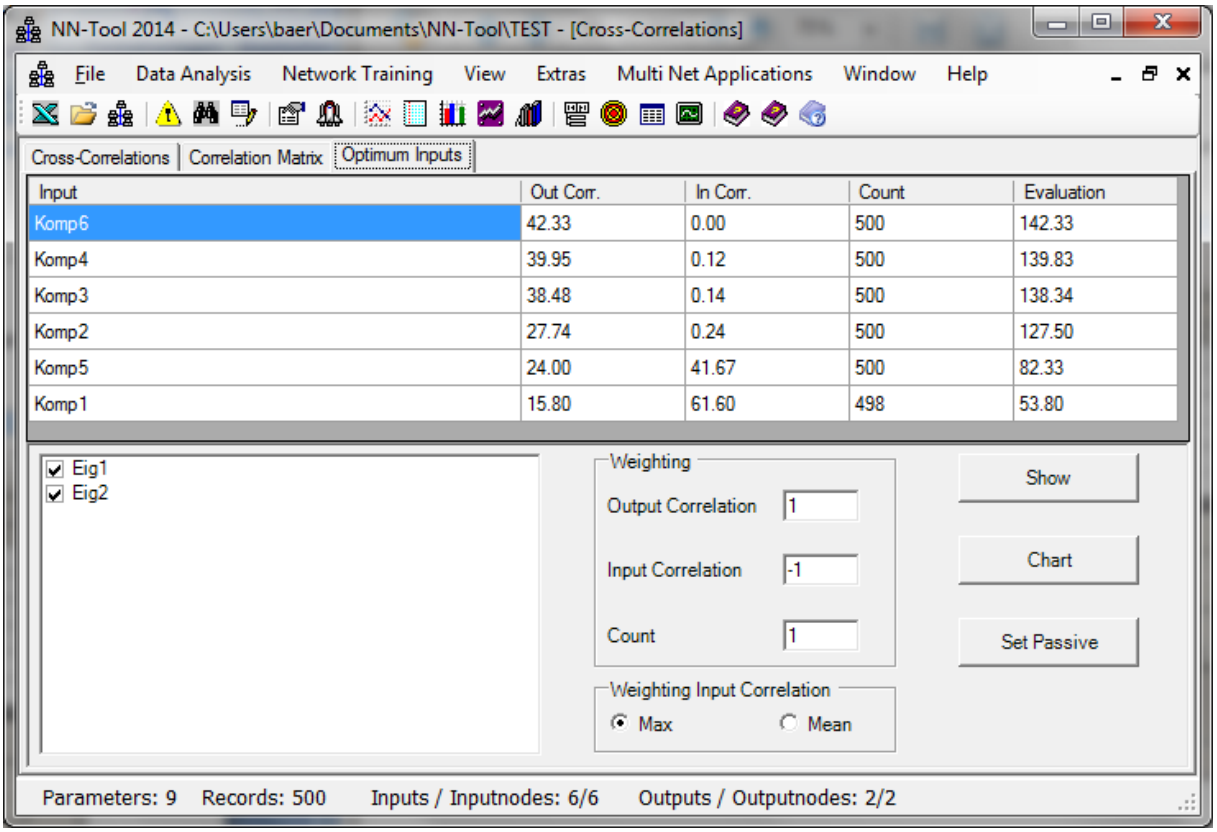


At first the correlation coefficient of each parameter is calculated with each other (**"Correlation Matrix"**). With m parameters and n records the computing effort is large-scaled proportional to $m^2 \cdot n$, i.e. generally high.
After the calculation the correlations of all other parameters to a selected parameter can be displayed. By the tab page "Correlation Matrix" the complete correlation matrix can be displayed and if necessary transferred to Excel.

Main application area: The correlation coefficient is a statistical measurement whether two parameters have much or little to do with each other. A correlation coefficient of +100% means that the both parameters are completely linear dependent and that the increase of the one always leads to an increase of the other. In case of –100% the both parameters are also completely linear dependent, but the one decreases when the other increases and reverse. A correlation coefficient near 0 indicates that there is at least no linear dependence between the parameters. This information can be used for the selection of input parameters. With two highly correlated input parameters (i.e. amount of the correlation coefficient near 100%) e.g. one of the both can be omitted, because the other provides already the essential information. This can be analyzed more exactly with the function "Optimum Inputs".

D. Optimum Inputs

The function "Optimum Inputs" requires that the input- and output parameters have been already determined. Then the following form supplies



for every input parameter a value how important this parameter is for prediction of the parameters selected in the lower area. The input parameter with the highest value is most important for the prediction. Three criterions are included in the evaluation:

1. **Out Corr**.: mean absolute correlation of the concerned parameter to the outputs. Should be as high as possible.

2. **In Corr**.: maximum or mean absolute correlation of an input to those other inputs, which are valued higher regarding the both other criterions. Background: input parameters are selected one after another according to their suitability for prediction. Another additional input parameter should have as little as possible to do with those already selected, but provide further independent information. Due to this reason the InCorr. of an input parameter should be as low as possible. Therefore it is entered negatively in the total value.

3. **Count**: Number of appearing of the parameters in the datasets. A suitable input parameter should occur in as many as possible records. Occurring in all records results in a number value of 100, occurring in 20% of the records results in a number value of 20.

The three criterions are weighted by the indicated weighting factors and result in the total value (“Evaluation”). The total value can also be negative. An input with a total value of –30 is less suitable than one with –10 or +5. The button "Show" makes a recalculation possible after replacement of the selected output parameters or changes of the weighting factors.

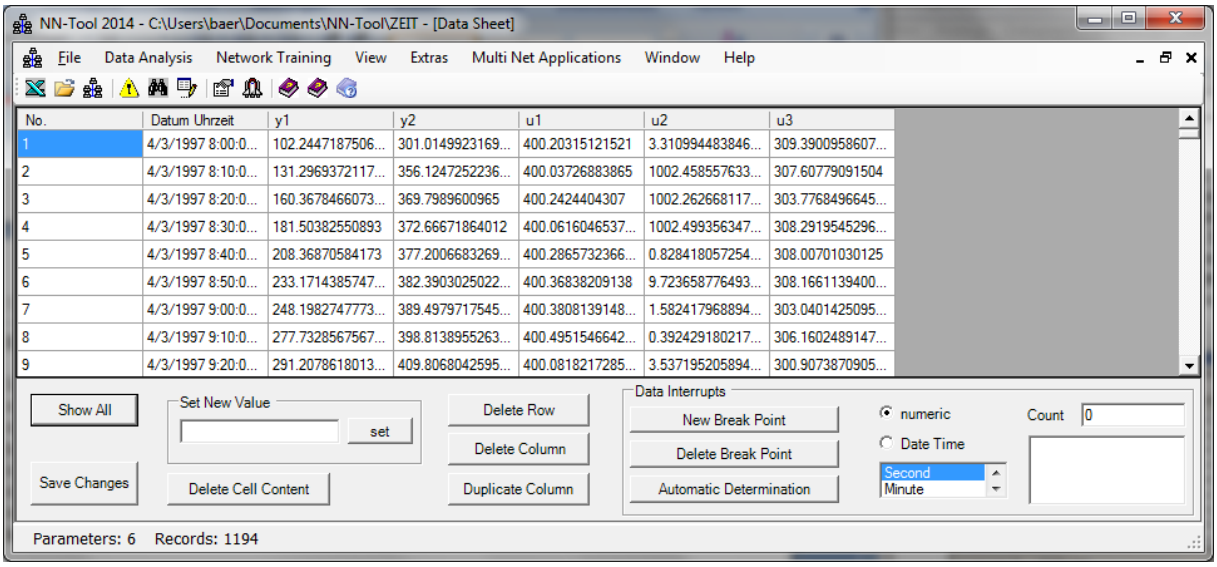
"Chart" supplies a corresponding diagram. With the button "Set passive“ you have the possibility to set every input passive that has too little ranking.

Main application area: In case of a lot of applications in the area of process modeling one has to do with a large number of input parameters, which are strongly correlated to each other (in case of recipe predictions this is mostly not so relevant). A unit is, for instance, equipped with a lot of pressure- or temperature sensors, which are all measuring similar values. This especially applies to time series. Then the same parameter occurs at different time delays. The function "Optimum Inputs" allows for a systematical selection of those from the large number of input parameters, which are especially suitable for prediction of outputs and are at the same time not correlated to each other as far as possible. That can also be used for the automatic delay time determination. For that you have to choose all possible time shifts in the time delay editor and determine the most suitable finally. However, please observe the higher requirement of time for calculation of the correlation matrix.

Important note:
In general, it is recommended to use the much more powerful “**Input Optimization”** procedure (see Chapter 6) instead of this functionality. The input optimization run also takes into account non-linear dependencies between input and output parameters.

E. Data Interrupts

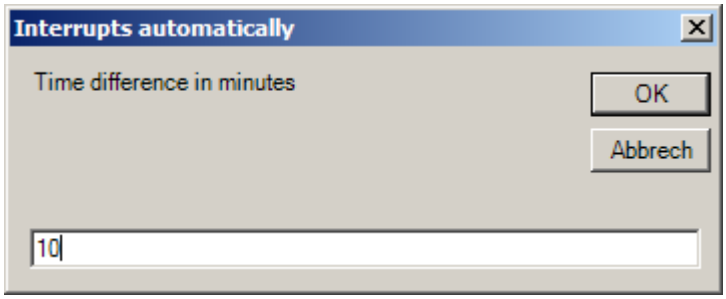
In Chapter 6, "Time series“, we had discussed the modeling of systems in which temporal shifts occur between the inputs and outputs. Using the menu item „Time Delays“ in the menu data analysis, we had provided parameter columns with temporal shifts. Here we had made, however, an improper simplification. It should be noted that the records in this example are no coherent data stream, but that in the meantime interrupts of the data stream arose by standstills of the plant. **After the data analysis and before the time delays are assigned** these interrupts have to be determined. Choose in the menu **Data Analysis** the menu point **Data Interrupts**:



It appears the already known window Data Sheet with the additional functionality of "Data Interrupts". The far right (initially empty) list shows the already specified interrupts using the record number. Interrupts can be specified manually with the buttons „New Break Point“ or „Delete Break Point“. If, for instance, an interrupt occurs between record 5 and record 6, please select these records with the mouse in the first column and press the button „New Break Point“. Enter a number of break points and then delete these again afterwards.

It is more efficient to determine the interrupt points by means of irregularities in the chosen column. Choose the column “**Datum Uhrzeit”** (Date Time**,** it does not matter here that this is a marker) by clicking in it and set the option **numeric / Date Time** on Date Time. It is useful to choose minutes for time measurement, because the regular measurement rate is 10 minutes. **Important:** "minutes" must not only be visible, but **must also have a blue background**.

Then push the button **“Automatic Determination”**. Then you will be asked to indicate the regular time difference in the chosen time measurement. In this example the measurement rate is 10 minutes:

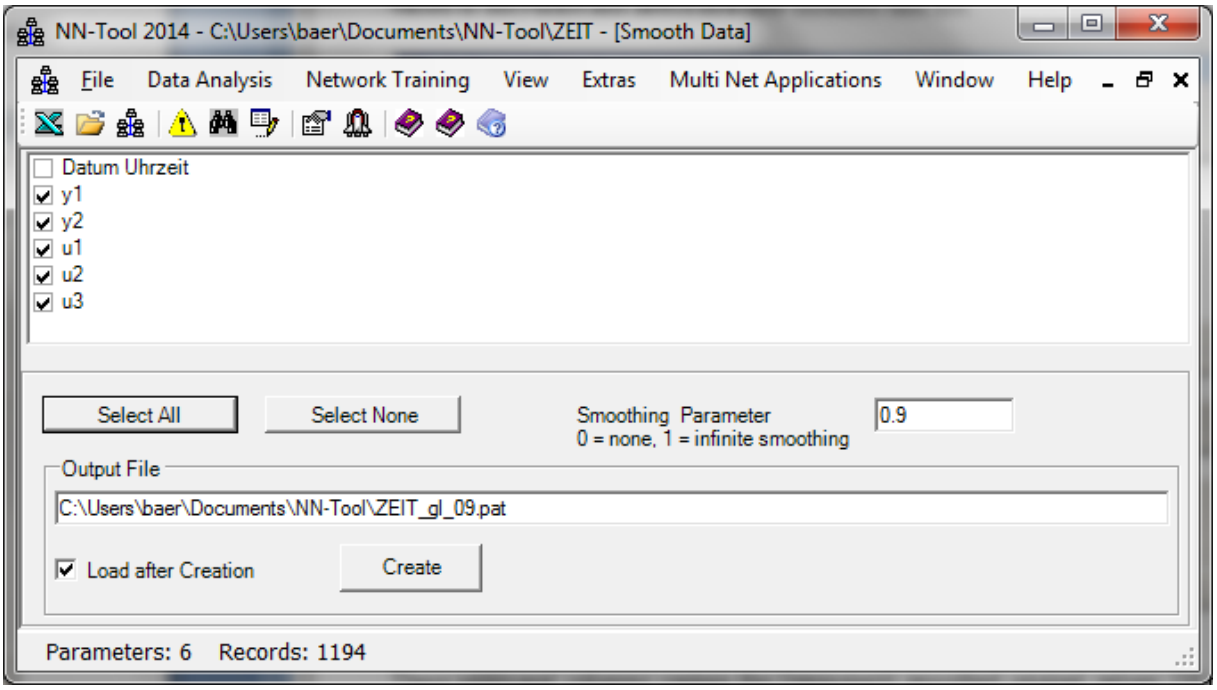


Following you will be asked for the maximum permissible variation of time measurement in % (e.g. due to manual timing). Enter here the value 30. That means: if two consecutive records are not apart less than 7 minutes (= 10 minutes – 30 %) and not more than 13 minutes, a data interrupt is assumed. In this example 4 break points (interrupts) have been found, listed and color-coded.

Now you can continue with the modeling as described in the chapter „Time Series“ by defining the „Time Delays“. The difference is that at the break points the time-shifted values can not be established. Thus, at these points the records are incomplete and are (correctly) not included in the modeling.

F. Smooth Data

This functionality in menu Data Analysis allows adding additional columns with smoothed values to the Patfile and save it as a new extended data file.



These additional columns contain the (temporary) smoothed original values. Of course this makes sense only for time series. The degree of smoothing is determined by the smoothing parameter; a value from 0 to 1 can be specified. The output file contains all the original columns and smoothed versions of the selected parameters (classifiers and markers can not be smoothed).

G. Append Data Files

With this menu item in main menu File, it is possible to join multiple .pat-files to one large .pat-file. Of course, the source files must have exactly the same parameters. This is especially interesting for plant modeling, when one data file is created per day and you want to use the data from several days to create a data model.

H. Reduce Datafile

This functionality in main menu File allows to select from a .pat-file only every nth record and to create a reduced .pat-file. This can be of particular interest for very large data sets (many thousands of records) with a low variation in successive records. The data reduction allows for a faster overview of the modeling process.

Annex 5: Outlier List

This component allows for the automatic determination of doubtful records. All records will be listed which have an especially large error between the predicted and the measured value (outlier). As a measure for the error the mean error on the test set is multiplied with an error factor (e.g. 2 or 3). Each record with a larger error than the double or triple of the mean error is determined by pushing the button "Calculate". Following the records can be tagged (button Select). Recently, a new reduced .pat-file can be created in which the tagged records are missing. With this new data file the network can be trained again.

Note:

Dynamic data sets are transformed to static data when the data is saved (this is necessary due to consistency reasons).

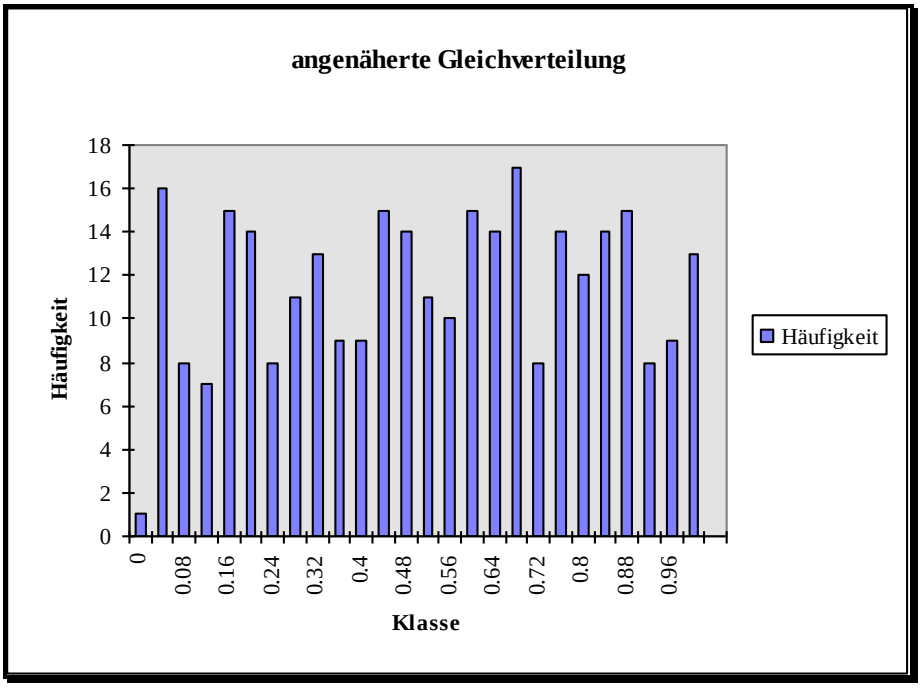
Annex 6: Validation Sets

A validation set is an independent dataset which is provided in a separate file (.vali-file) and has to be predicted by the network. The structure of a validation file has to correspond exactly to the structure of a .pat-file. Due to reasons of advisability one will fall back on the function for creating .pat-files, provided by NN-Tool under Excel. The prediction is executed by choosing the menu point **"Prediction (Validationset)"** in the menu "Network Training".
This mechanism can also be used for prediction of new datasets. Then any values (e.g. 0) are entered in the outputs (empty values are not allowed). After prediction the corresponding single values can be written and analyzed in an Excel-file by the button Excel.

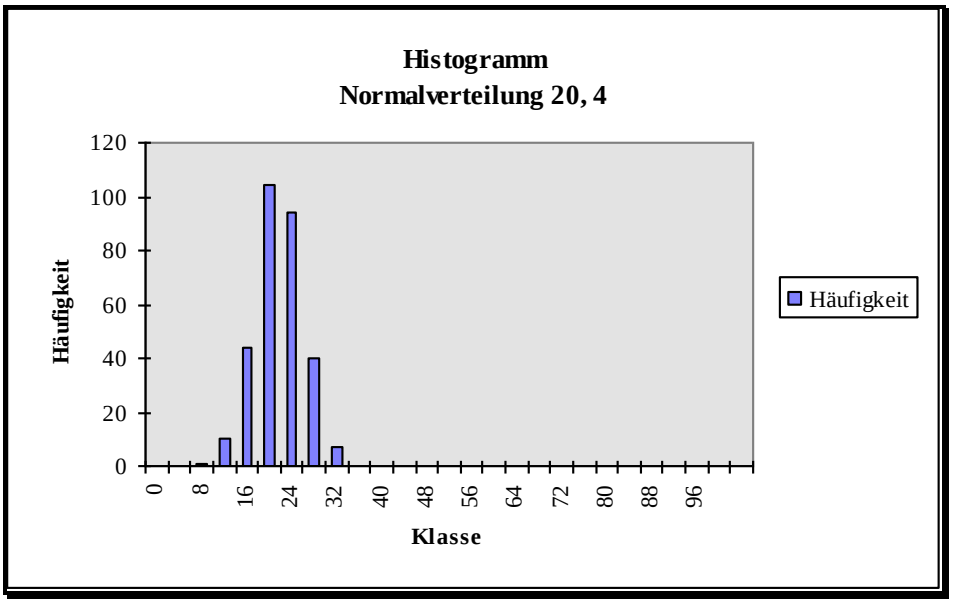
Annex 7: Nonlinear Scaling Transformations

A7a) Meaning of statistical distribution of input- and output values

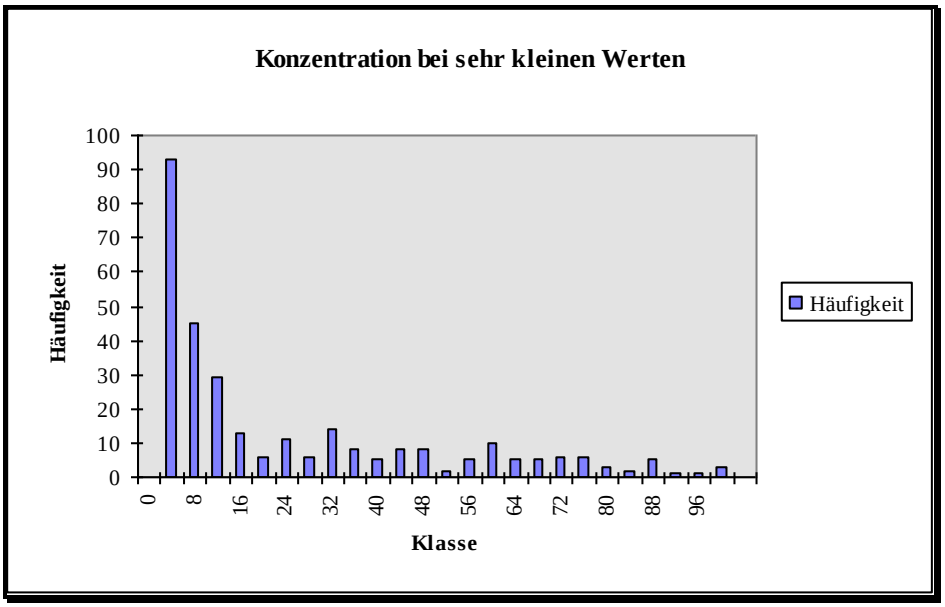
In order to enable a neural network to learn optimally the mathematical connection between different parameters, these parameter should be, if possible, distributed uniform over their range (picture 1 approximate uniform distribution).



In a lot of application cases single parameters (measured quantities) are, however, not at all distributed uniform, but the measurements concentrate on certain points. The following diagram (picture 2) shows a normal distribution with mean value 20 and standard deviation 4:



Beside the normal distribution a concentration often occurs with very small values (picture 3):



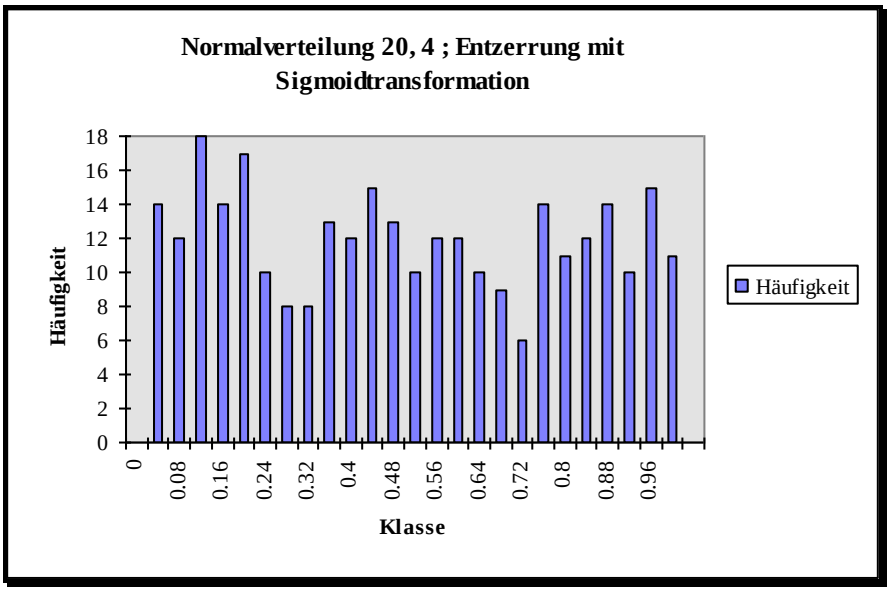
While the concentration of data points is partially still tolerable in the input parameters (here the NN itself can provide for an equalization), such output parameters should be retransformed at first in new, more uniformly distributed values. Presently, beside the linear transformation (standard case) the both following transformations are available in NN-Tool:

1. Sigmoid-transformation: $T_S(x) = \dfrac{1}{1+e^{-a(x-m_x)}}$
2. Logarithmic transformation: $T_L(x) = \dfrac{\log(x) - \log(min)}{\log(max) - \log(min)}$

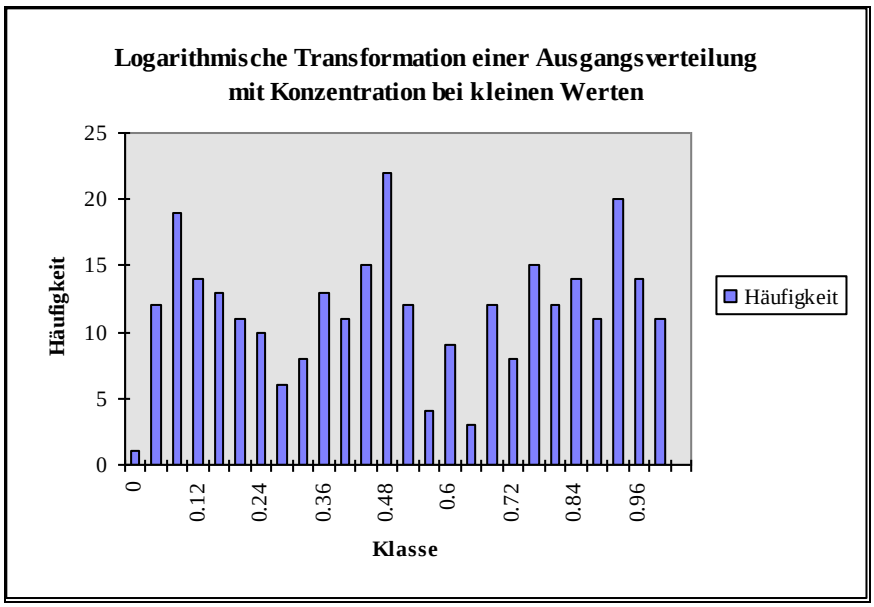
Min, max and m_x mean the minimum and maximum values of the value x as well as its mean value. For a is valid: $a = \dfrac{1.6}{\text{standardde viation}(x)}$

The value 1.6 can be explained theoretically. The theory supplies: $\dfrac{4}{\sqrt{2\pi}} \cong 1.5958$

After application of the Sigmoid-transformation on the normal distribution of picture 2 the following distribution (picture 4) results for the transformed value:



i.e. the new value is considerably more uniformly distributed over its range.
For the distribution of picture 3 the following results after application of logarithmic transformation (picture 5):



Also this value is now distributed more uniformly over its range.

A7b) Application in NN-Tool

The transformations can be set manually in the data analysis window or in the window histogram. However, the question arises when which transformation is to be applied.

Automatic Application:
Use the buttons **„Auto-Inputs“** and **„Auto-Outputs“** in the window „Data Analysis“ to determine the suitable transformations. For this purpose, NN-Tool calculates two criteria:

$$K_1 = \frac{\text{Max} - \text{Min}}{\text{Stddev}}$$

$$K_2 = \frac{\text{Mittel} - \text{Min}}{\text{Median} - \text{Min}}$$

The median of a size x is a value for which 50 % of the values of x are higher and 50 % are smaller. The median is different from the mean value (that can be realized by the example of the financial circumstances of a billionaire and his three cleaning ladies).

NN-Tool sets the transformations, if one of the following conditions is met:

$K_1 > 5.7 \Rightarrow$ apply Sigmoid-transformation.
$K_2 > 2.5 \Rightarrow$ apply logarithmic transformation.

Note of the author: There is not much experience with this option yet. After first tests this seems to be a very useful option with the output parameters. In case of input parameters the use of non-linear transformations is more questionable, especially with recipe property relations.

Annex 8: Integration of previous Knowledge / Input Output Assignment

The window Data Analysis offers the following options to integrate previous knowledge:

Tab Page IO-Assignment: Allows the assignment of input to output parameters. By default, each active input parameter is used for modeling each output parameter. Using this feature, NN-Tool can be instructed to ignore certain inputs with certain outputs, or to allow only pure monotonous connections. A disabled input does not affect the number of complete records for an output parameter. I.e. if a rarely measured input parameter for a given output is disabled, it does not reduce the total number of records for this output.

Available Records: This information in the appropriate column of the data analysis window displays in advance how many complete records are available for the learning process of each output parameter. Where appropriate, some rarely measured inputs should be set passive, or (using the IO-Assignment) should be switched off for certain outputs. With these actions you can increase the number of complete records.

Annex 9: Tab Page Training Parameters / Crossvalidation

With the tab „**Training Parameters**“ in the Data Analysis window additional options for learning can be set. These settings are particularly advantageously in critical applications where the ratio of the number of complete records on the number of input parameters is small (e.g. smaller 3). The most important measure is certainly the distribution of records. Here crossvalidation options are of special importance for critical applications.

Distribution of Records: In some applications, the quality of the resulting network models depends sensitively on which records have been used for training (training set) and which have been used for testing. This is particularly the case when only a few records are available or the records show a high correlation to each other (e.g., for time series applications). For this reason, the following options for the portioning are available:

- **Options: every nth Record in Test set / Coherent Test Set:** indicates how the records of the testset are to be chosen from the available records for an output node. For test series the option "Every nth Record in Test Set" is more useful, for time series the option "Coherent Test Set" should be preferred, because of the typically high correlation of neighboring records. NN-Tool chooses the corresponding option automatically, i.e. also here normally no change is necessary.
- **Specific Distribution:** Allows detailed assignment of every record to the test set or the training set respectively by using the tab page **"Training Set / Test Set Allocation"**. Furthermore it is possible to assign a weight to every record. This weight indicates how often a record will be used (0 = record ignored).
- **Crossvalidation:** Allows using of multiple, dynamically created test sets for every output. This option reduces the dependence of the resulting network structure from the selection of a specific test set. If for example the value of the field n-fold Crossvalidation is 5 then 5 partitions of learning and test set are made for each output parameter. For each of these partitions, the entire test program in terms of internal nodes and the training steps is executed automatically. Then, each output node will be trained with the optimal network structure on all records. Finally, the individual sub-networks will be joined to the overall network. **Note:** A specific weighting of records in tab page "Training Set / Test Set Allocation" will also be taken into account using crossvalidation. For the dynamic allocation of the records four different strategies are available:
  - **n-fold Crossvalidation Cyclic**: With this option and n = 5, the first dynamic test set contains the following complete records: 1, 6, 11, .. the second test set consists of records 2, 7, 12, .. etc. This partitioning is generally especially suitable for uncorrelated data series.
  - **n-fold Crossvalidation in Blocks**: In this option, and n = 5, the first dynamic test set contains the first fifth of the records, the second test set contains the second fifth, etc. This partitioning is generally particularly suited for time-dependent data (time series, see next example).
  - **n-fold Crossvalidation Random**: In this option, and n = 5, each of the five test sets consists of records which are assigned to the test set with a probability specified in the field %-Fraction. In this mode, no final assessment of the test set accuracy can be carried out (only training set).

- **Crossvalidation Leave-One-Out**: The extreme case "Leave-One-Out" creates as many different test sets as there are complete records for the output under consideration. In this case every test set consists of only one record (no correlation information is possible during training runs). This mode is by far the most expensive in terms of calculation time, but can - at least for uncorrelated data – achieve the highest model accuracy. Especially for small data sets of (hopefully) uncorrelated test series suitable. In time series, this option - because of the generally high correlation of records among each other - is not recommended.

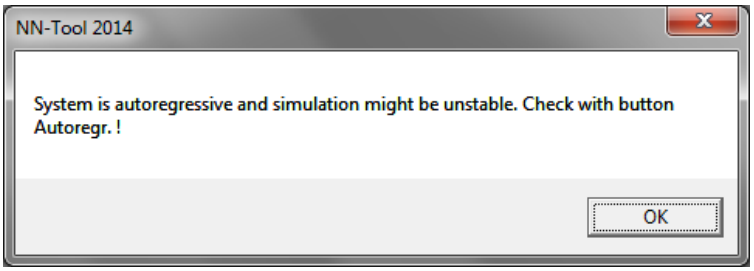
The following option is often useful for correlated data:

- **Error Rating in %:** As already mentioned NN-Tool tests for each output parameter singly the optimum net configuration with different numbers of training steps and inner nodes. Finally the net is used which achieves the best prediction precision on the test set. Normally for that the mean relative errors of the single nets are compared. With this option punishment terms can be additionally put on networks having a lot of inner nodes. A value of $x > 0$ with this option means that the mean relative error of a fraction net with n nodes is multiplied by a factor of $1+ (nx / 100)$. This will be used for the network choice. This option is of importance, when a high correlation of testing records with corresponding training records has to be assumed. In this case the test set error itself does not offer a sufficient basis for assessment.

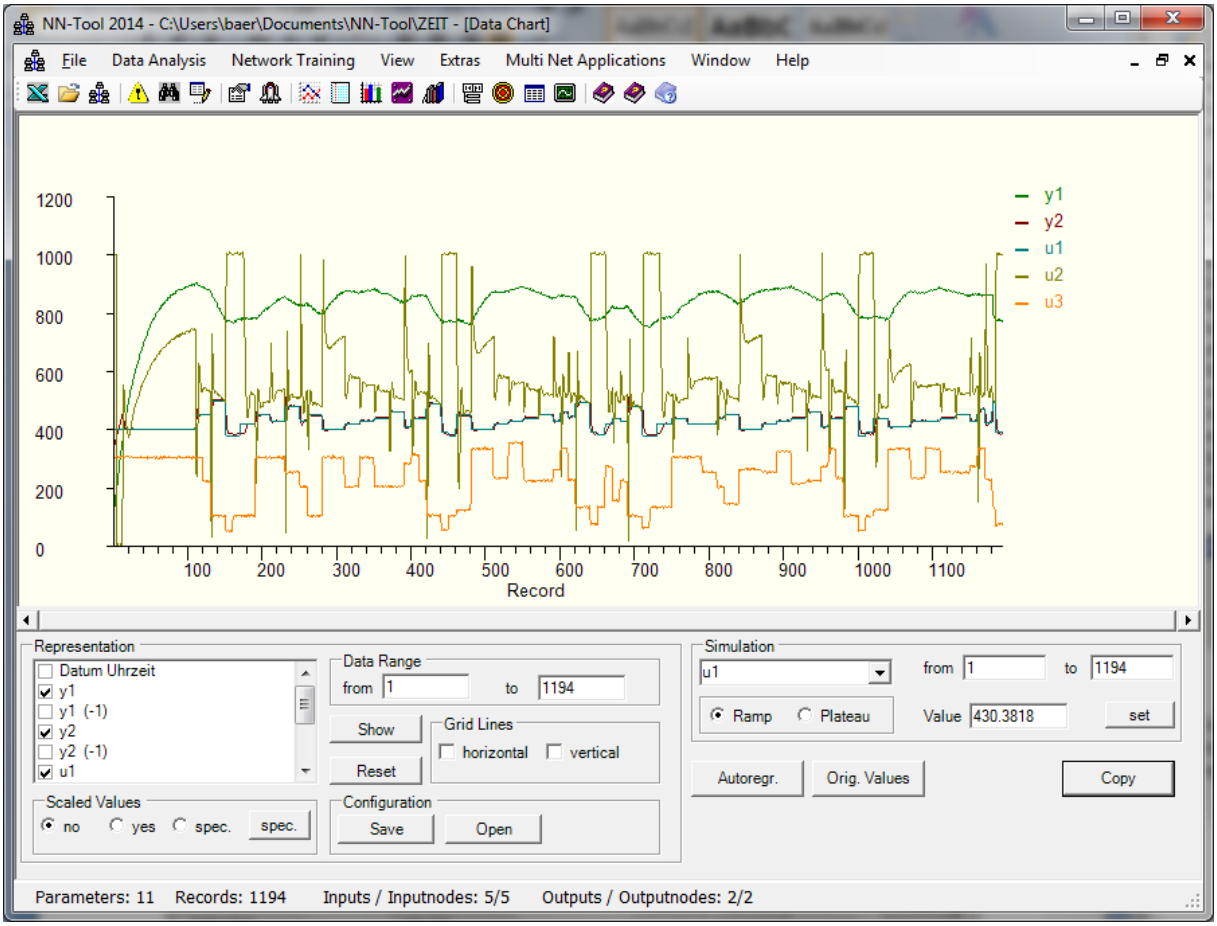
Annex 10: Dynamic Simulation

This application in the main menu "Extras" allows the simulation of time series. It is quite analogous to the window "Data Chart" (Annex 4). Please make yourself familiar with this application first. Then load (or create) the network **zeit.dsc** as described in chapter 6. The application „Zeit“ is a so-called autoregressive model. **Autoregressive** means that at least one of the outputs also occurs as an input, then of course at an earlier time. I.e. the modeling of a parameter is based (partly) on the same parameter (hence the "Auto"). The system „Zeit“ would be e.g. non-autoregressive, if the inputs would consist of the u-parameters only and the output only of the y-parameters. Autoregressive is not necessarily identical to the occurrence of time delays. Autoregressive models always use time delays, but not every model with time delays is an autoregressive one.

Why is this characterization important? If simulations are carried out using an autoregressive model, you will usually predict the outcome for certain specified data histories of input variables. If more than one step in the future has to be predicted, the model must be applied repeatedly. So the outputs in the previous forecast step (here y) must be used as inputs in the next step. I.e. from the second step, the simulation is, in contrast to the original model, no longer based solely on measured data. The further one predicts in the future, the more often the calculated values, which may include model error, were used. This means that autoregressive models are dynamic systems in the mathematical sense, while non-autoregressive models are simple functions. Specifically autoregressive models bear the risk that the model error is more and more increasing, the further you predict in the future. In this case, the simulation is described as unstable. This effect can occur variably, you should at least be aware of this possibility. Now open the application "Dynamic Simulation". First you will be asked for a file, because the simulation can be made in principle with each data file having the .pat-file format (for example, when new data sets of the process are available). Select zeit.pat. If the system is autoregressive, the following alert indicates the possibility of an unstable simulation.

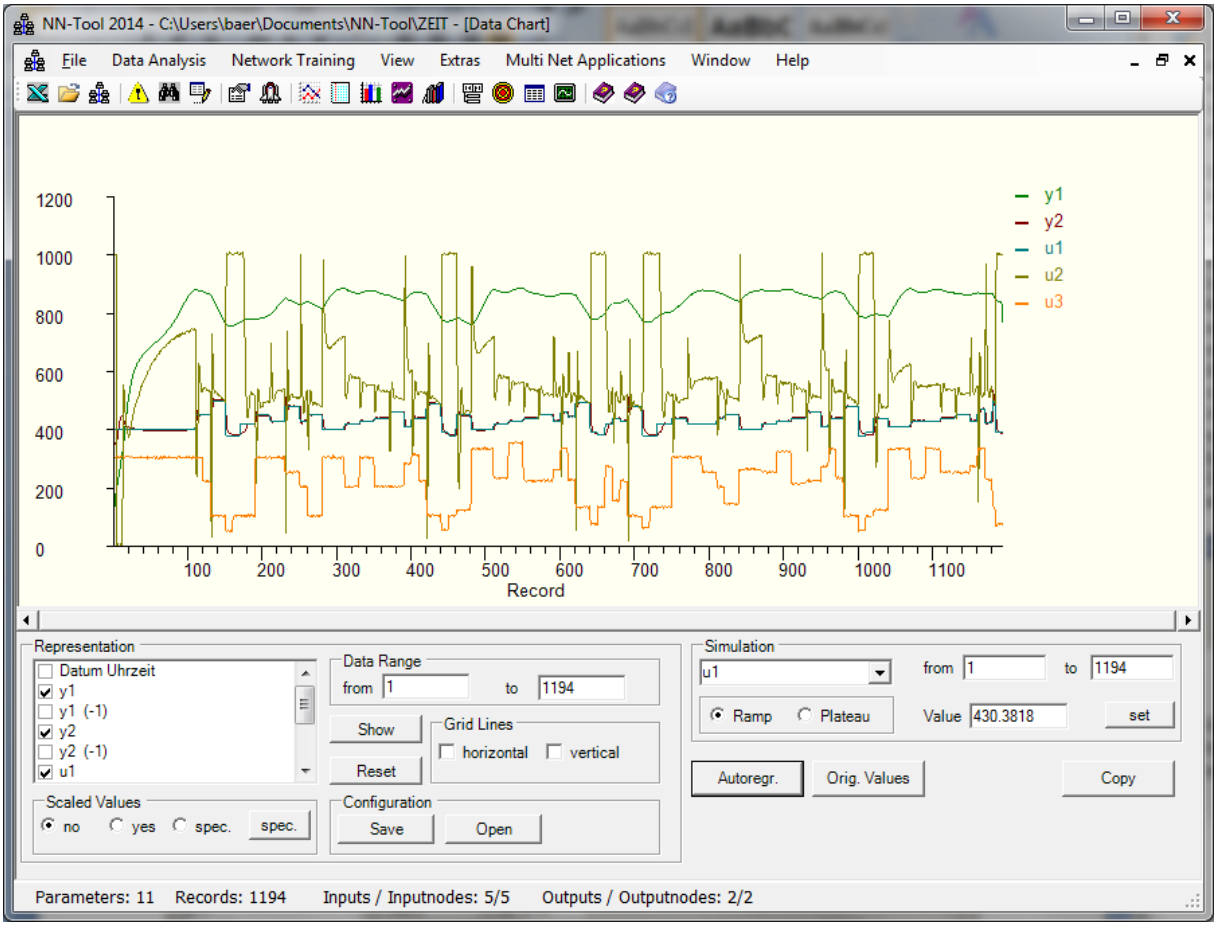


In the case of autoregressive models a corresponding stability test should be carried out with the button „**Autoregr**.“ (see below). After confirming the message, the following window appears:

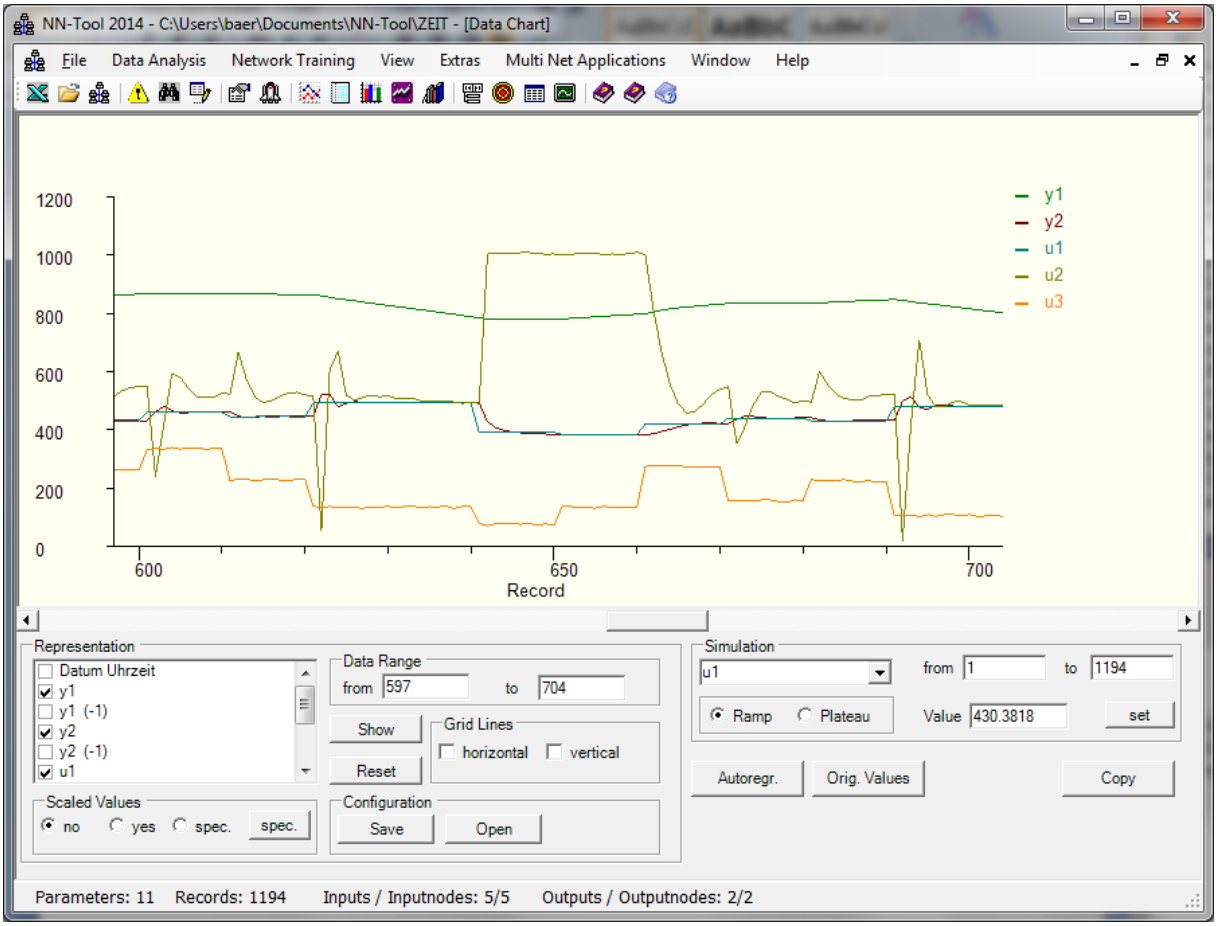


The window is, with few exceptions, identical to the window Data Chart. However for the output variables not the measured values are presented, but predicted (calculated) values on the basis of the input variables. These values are identical with the predicted values of the „Line Diagrams“. Note that this calculation is non-autoregressive, because the necessary input variables here are completely available as measurements. As long as possible, you should always calculate non-autoregressive and use measurements as input variables.

With the button **"Autoregr."** now the complete time series can be simulated autoregressive:



It changes the course of y1 a little bit, especially in the first records, which correspond to the start-up procedure of the process and are represented in the data only once. This is also the reason for the differences between training and test set. The value of y2 changed almost not at all. To be better able to see the effect, you should possibly switch-off the inputs from the graphic and set the scaling to **„spec.**“. With the button "**Orig. Values**" the data can be set back to the original values, i.e. to the original inputs and to non-autoregressive predicted outputs. Here the comparison points out for a sufficiently stable situation even for a long time simulation. Now, using the right mouse button cut out the data range between record number 600 and 700.



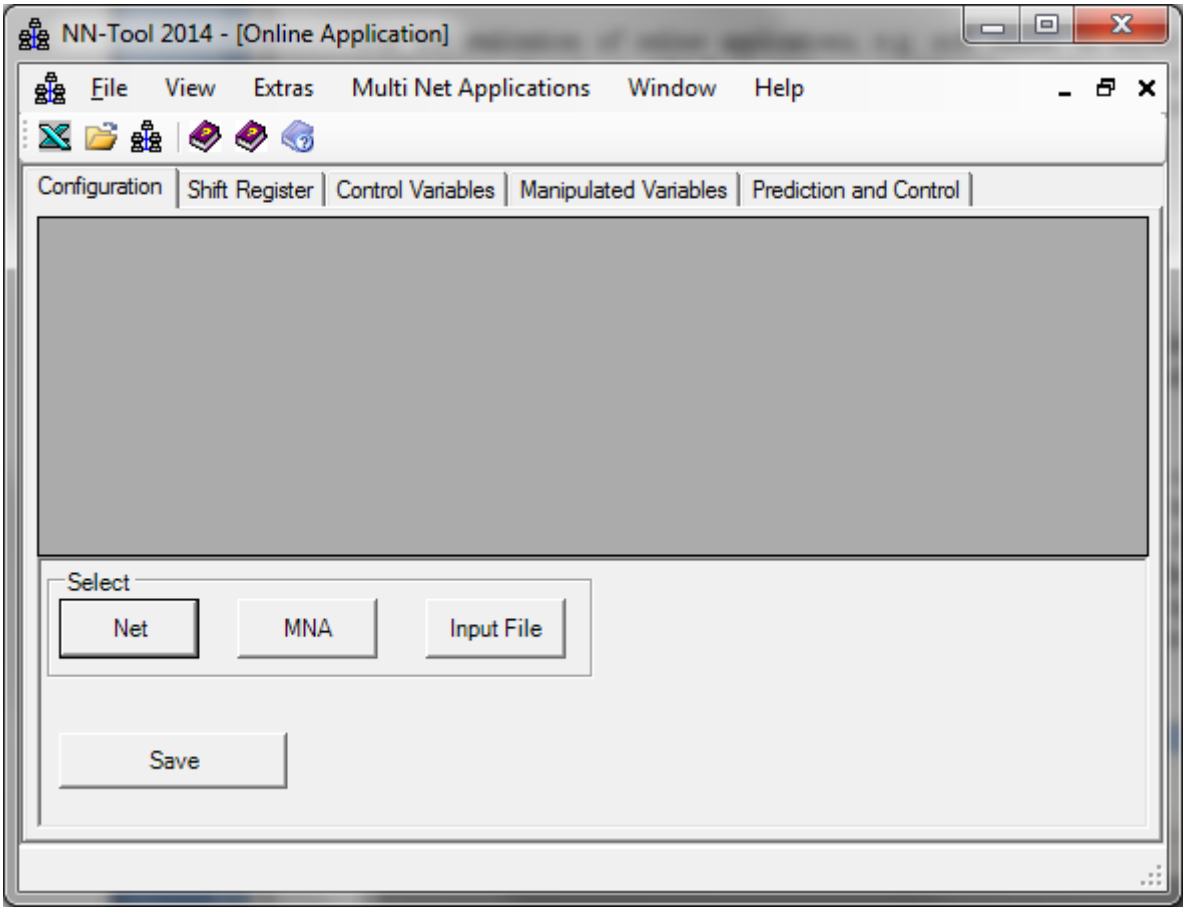
Now you can define new value courses for the input parameters of the model and the appropriate courses of output variables will be calculated.
Note: The value specification will be made always for the zero delay time values, even if they do not contribute to the modeling (as here). The corresponding shifted values (e.g. u1 (-1)) will be provided automatically. The specification can be made using a from-to-range and a value for the selected parameter. For the course you may choose a ramp or a plateau. The specifications can also be made using the left mouse button. The button **"Orig. Values"** sets back the values.

Annex 11: Online Applications

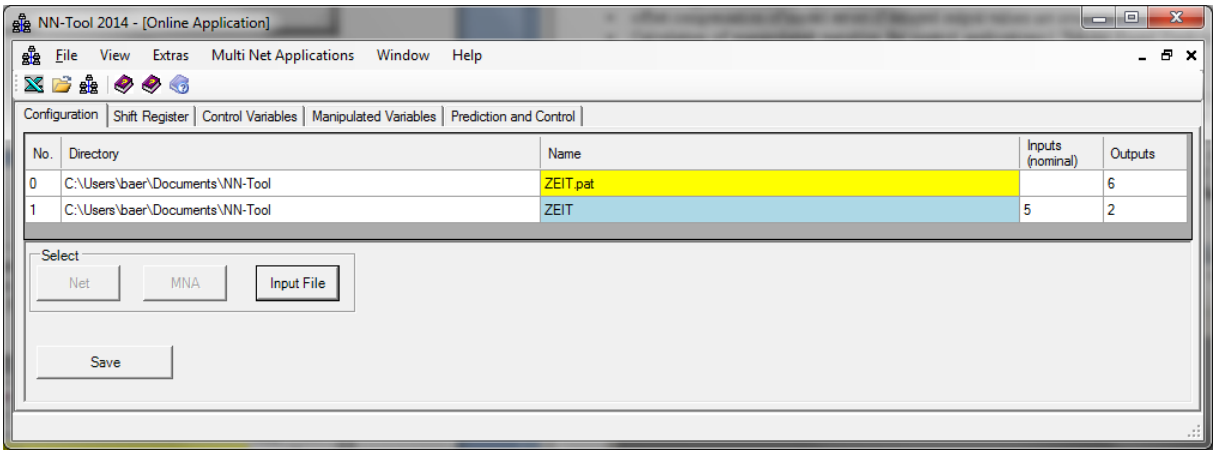
For the realization of online applications, e.g. soft sensor or model predictive control applications, based on NN-Tool process models, the additional software component „**NNControlServer**" (ActiveX Library) is available. This component provides, among others, the following functionality:

- integration in arbitrary programs
- loading of NN-Tool applications
- prediction of individual records for defined time intervals in the future
- providing of shift registers for time delays
- smoothing filter for noisy input data
- offset compensation of model errors if delayed output values are available
- Calculation of manipulated variables for control applications ("Model-Based Predictive Control")

The application is not based directly on a NN-Tool network, but on a so-called .mna-File (see Annex 16). The conversion and adaptation of a NN-Tool network for the specific needs of online applications can be done in NN-Tool itself. In this environment the functionality can also be tested through simulations. The procedure is described in the following on the example of the application „Zeit“ (see Section 8 Time Series). Build the application „Zeit“ as described in Chapter 8. On the main Menu "**Online-Applications**," choose the menu **"New Online Application":**

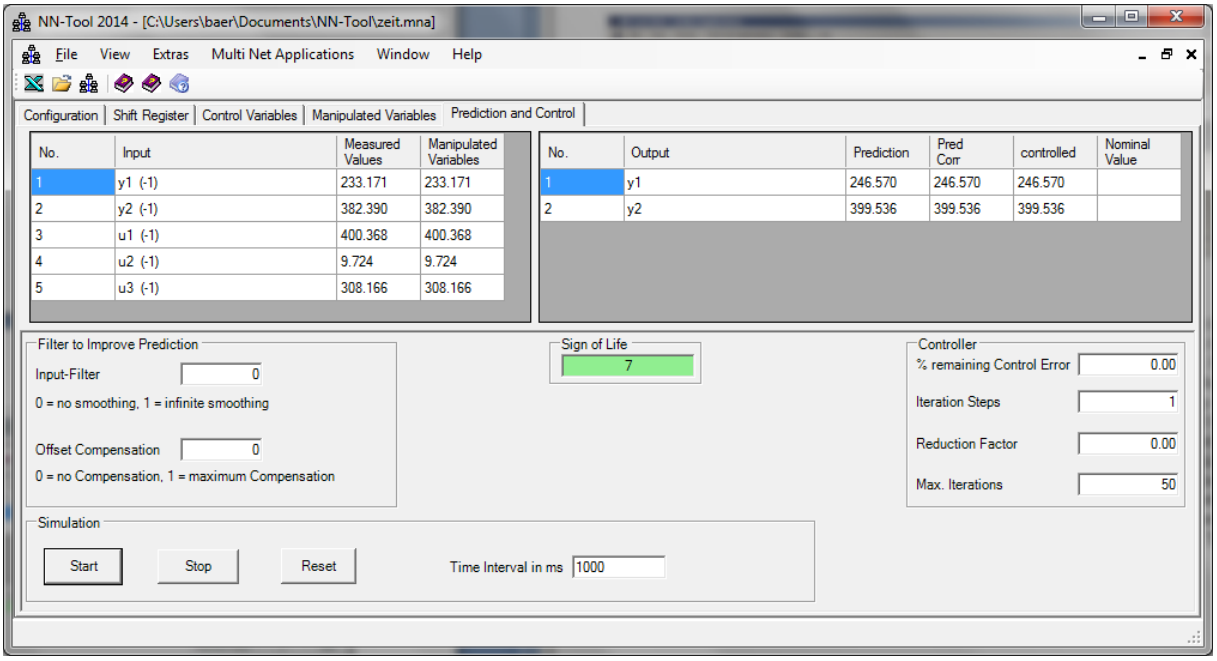


An online application is based on a multi net application (or more accurately is an extension). If such an application is not yet available, a new multi network application can be configured that consists of a single NN-Tool network. Press the button "Net", select the NN-Tool network "ZEIT.dsc



and press Save. Save the application under the name "Zeit.mna".

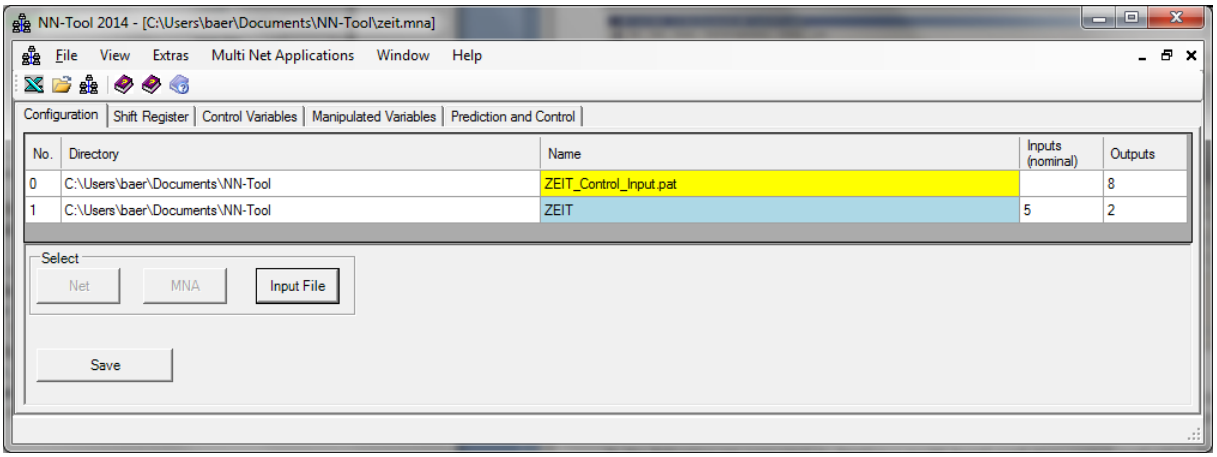
This is a minimal configuration and the network for a mere prediction application under the name "**Zeit.mna**" is ready. The configuration requires that the record transferred from/to the online application has exactly the format of the file specified in line "0" ("**Input File**"). By default, the data format of the associated .pat-file is used. If the application uses time delays (as here), a corresponding shift register is also provided (see tab "Shift Register"). Now go to the tab **"Prediction and Control"** and press the **"Start"** button to start the **simulation of the online application**. These records are not coming from an external application ("simulation"), but are read one by one from the file in line 0 ("inputfile"), transferred to the online component, and the predicted values are written in the file "<Application> _output.txt" (here "Zeit_output.txt"). In the subsequent online application the transfer of data occurs then with the calling program, of course. The left side of the picture shows in column "**Measured Values**", those values, which are supplied to the component by the calling program. The values on the right side with the name '**Pred. Corr.**" will be returned from the component.



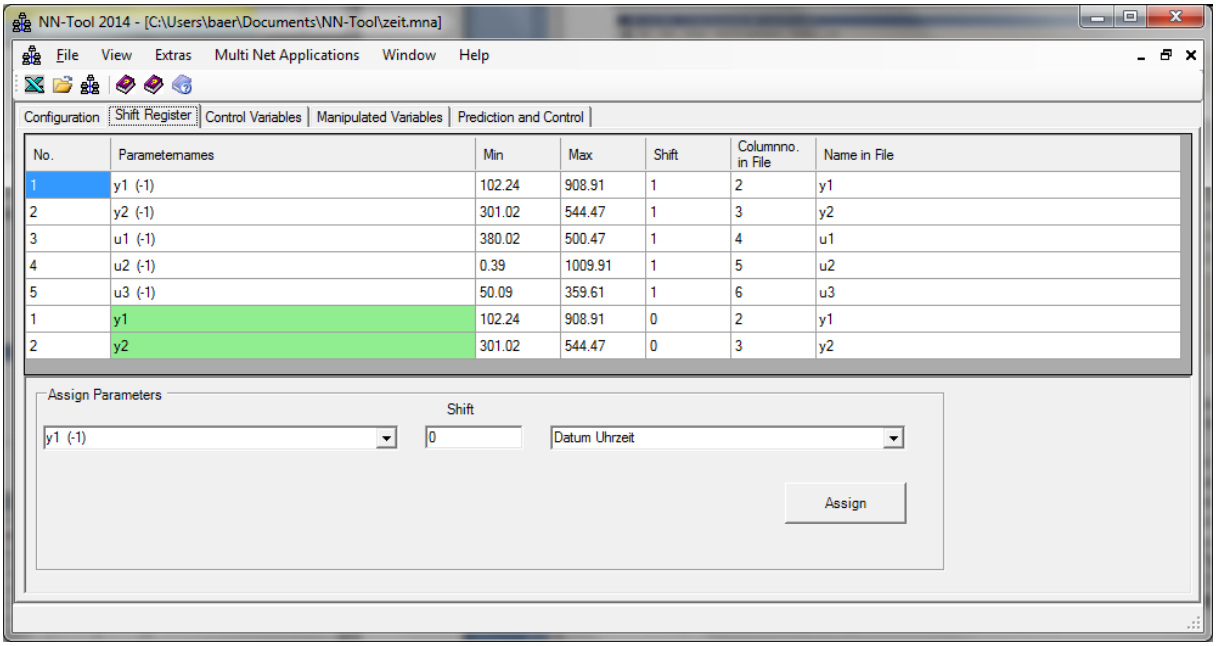
"**Sign of Life**" provides a number greater than zero (1 to 10,000) if the data transfer and the evaluation were successful, or a negative value (error code) and a red box if an error occurred.

In the following we now want to develop a model based predictive control application from this simple prediction application. Press the button "**Stop**", then "**Reset**". First we need to ensure for the control application that now also target values (nominal values, setpoints) for the outputs of the network are provided. These values (with the other data) must also be transferred to the online component. For this we need a correspondingly expanded input file that provides these setpoints. Change to the **„Configuration“** tab and press the button "**Input File**". Select the file **"ZEIT_Control_Input.pat“** from the folder "My Documents\NN-Tool". Compared to "ZEIT.pat" the new file has two additional columns "**Soll_y1**" and "**Soll_y2**", containing target values for y1 and y2.

Hint: The file is using decimal comma. If you are using decimal point, please use an editor to replace all commas “,” in this file by dots “.”.

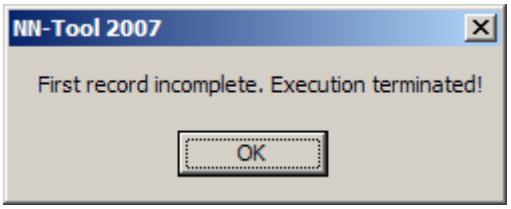


Since a new input file has been assigned to the system, it is no more automatically ensured that the inputs are correctly supplied with values. Go to the tab "**Shift Register**":



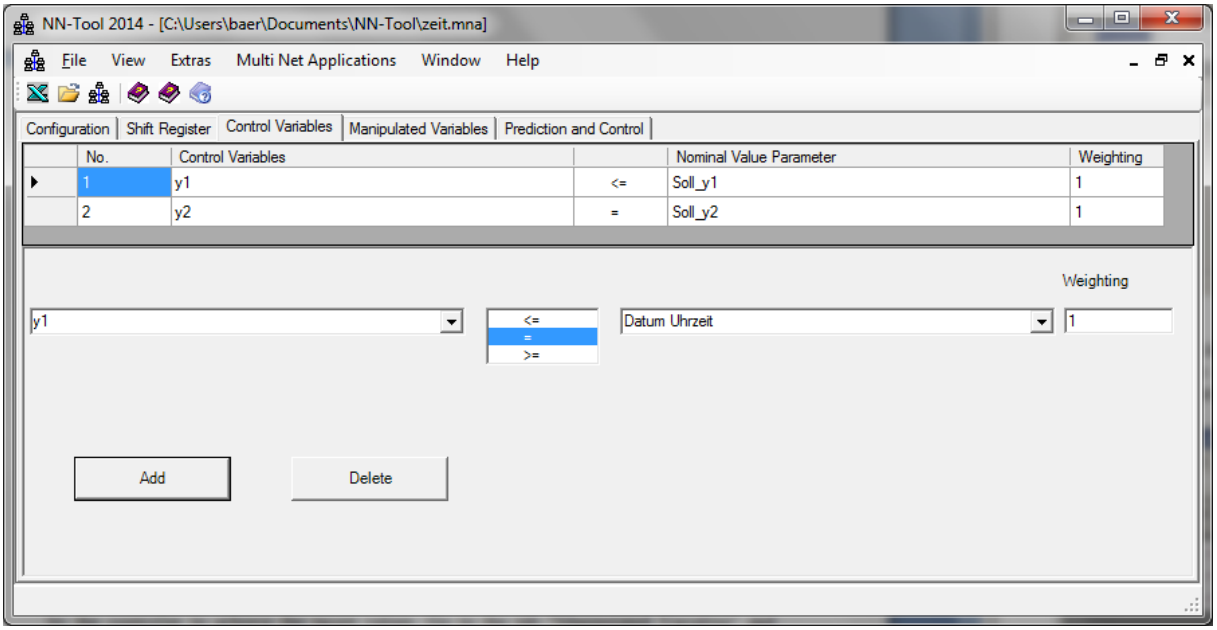
As you can see from the parameter names and the shifts (i.e., time shifts), NN-Tool has done the correct assignments automatically.

Note: This does not work always; in these cases, the assignment has to be done manually. The assignment can be done with the two list fields and the button "Assign“. If no correct assignment is made, the simulation provides the error message:

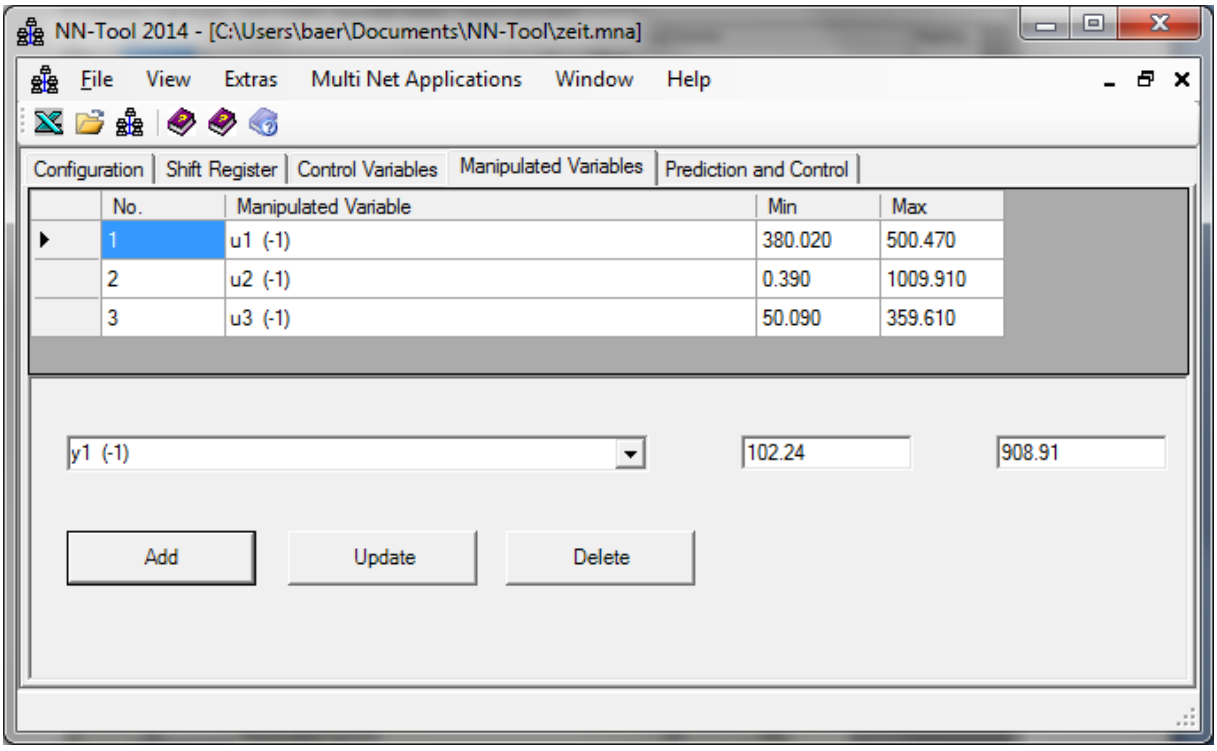


However, the assignment is correct here.

Now go to the tab "**Control Variables**" and specify "**y1 <= Soll_y1**" and "**y2 = Soll_y2".**



The next step is to define the manipulated variables, i.e. the parameters that may be changed by the controller to achieve the target values. Go to the tab "**Manipulated Variables**" and define the control variables u1, u2, u3 as shown:

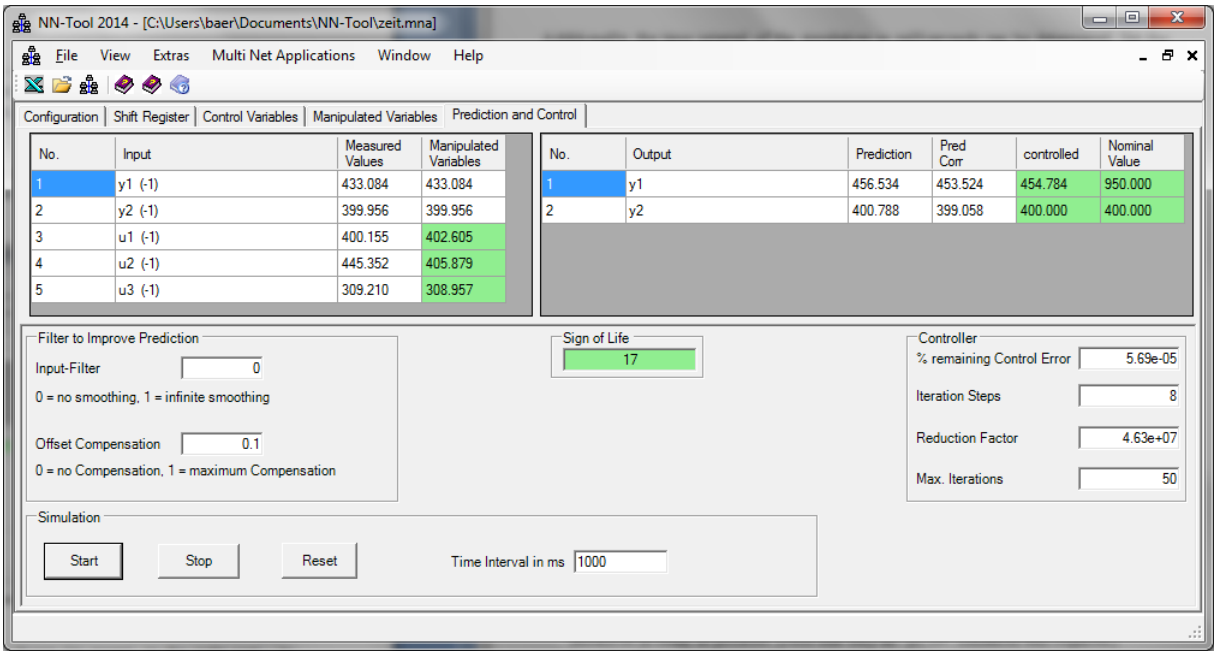


Change to the „**Configuration**“ tab and save the application under the name **„Zeit.mna“**. Now go to the tab **"Prediction and Control".** Here you can adjust the following additional options:

- **Input Filter**: first-order filter to smooth the input data.
- **Offset Compensation**: enables correction of the model error by parallel evaluation of the model.

- **Max. Iterations**: The manipulated variables are calculated in an iterative process. The greater the value the more accurate the calculation, the longer it lasts.

Additionally, the time interval of the simulation in milliseconds can be determined. Set the offset compensation to the value 0.1 , otherwise retain the default settings, **save the configuration** and press Start.



On the left the calculated manipulated variables are shown (in green), in the right part - in addition to the prediction values of the model - the corrected predictions, which now differ from the model's prediction due to the offset compensation, are displayed. In the column "controlled" those output values are displayed that are to be expected in the next time step according to the model, if the manipulated variables are used in the process. In addition, the following information is provided which describes the control accuracy:

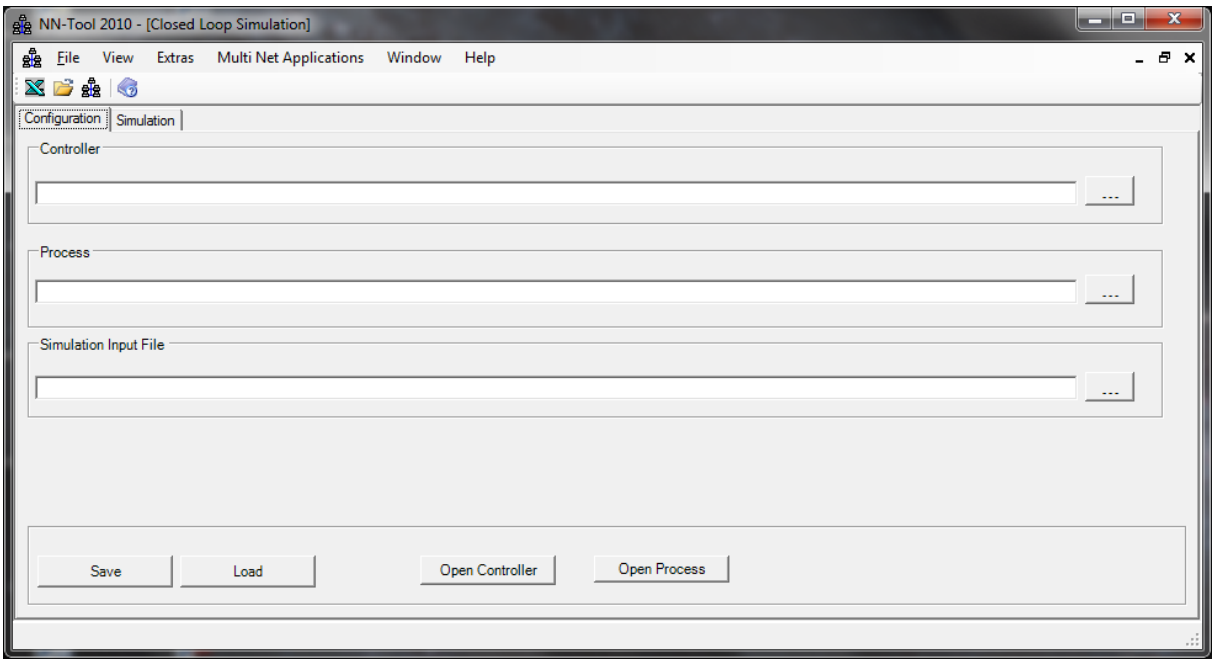
- **% remaining Control Error**: Indicates how accurate the target values were achieved. Should be as small as possible. (Note that only an "y1 <=" condition was required.)
- **Iteration Steps**: the number of required calculation steps.
- **Reduction Factor**: The factor by what the original control deviation could be reduced. Should be as large as possible.

The simulation now writes the original records and in addition, the corrected predictions and the manipulated variables in the file **„<application>_Output.txt**"(here "**Zeit_Output.txt**"). This file may not be blocked by other applications (possibly restart NN-Tool).

Note: A .**NET runtime library** is available for this NN tool component in order to couple the application with external applications (especially process control systems) (see product overview).

Annex 12: Closed Loop Simulation

The application **"Closed Loop Simulation"** in the main menu "**Online Applications**" provides a closed-loop test of a model-based controller (see previous Annex 11: Online Applications):



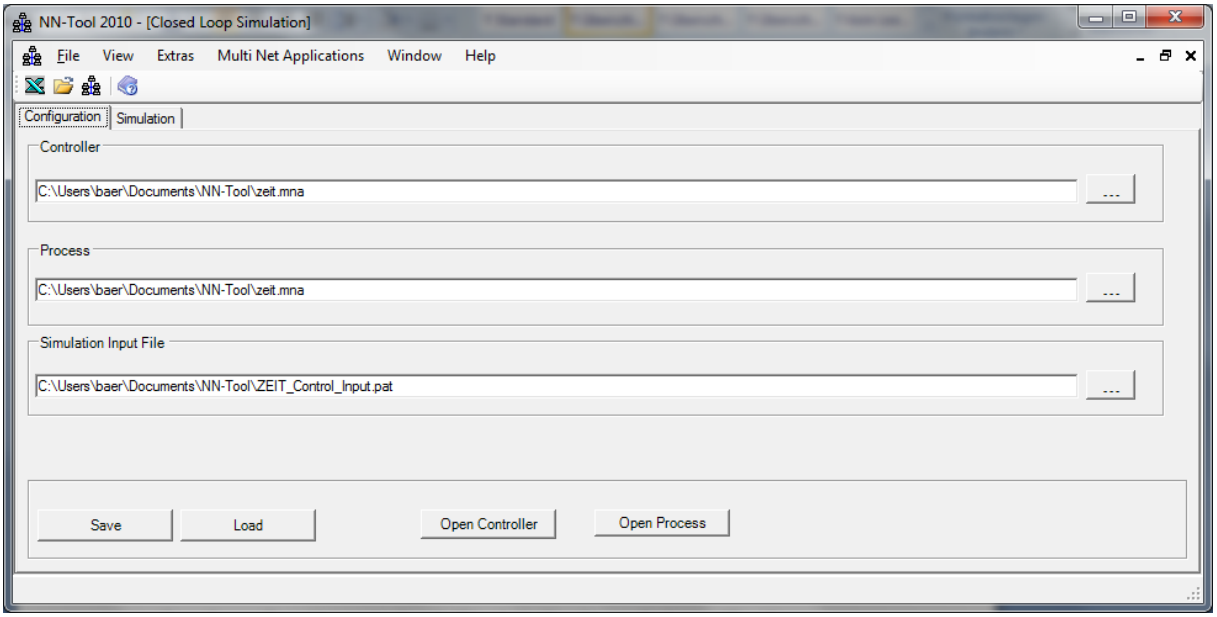
For the controller ("Controller") select the model-based controller "**zeit.mna**" designed in the previous chapter. Select the same application for the simulation of the process ("Process").

Notes:

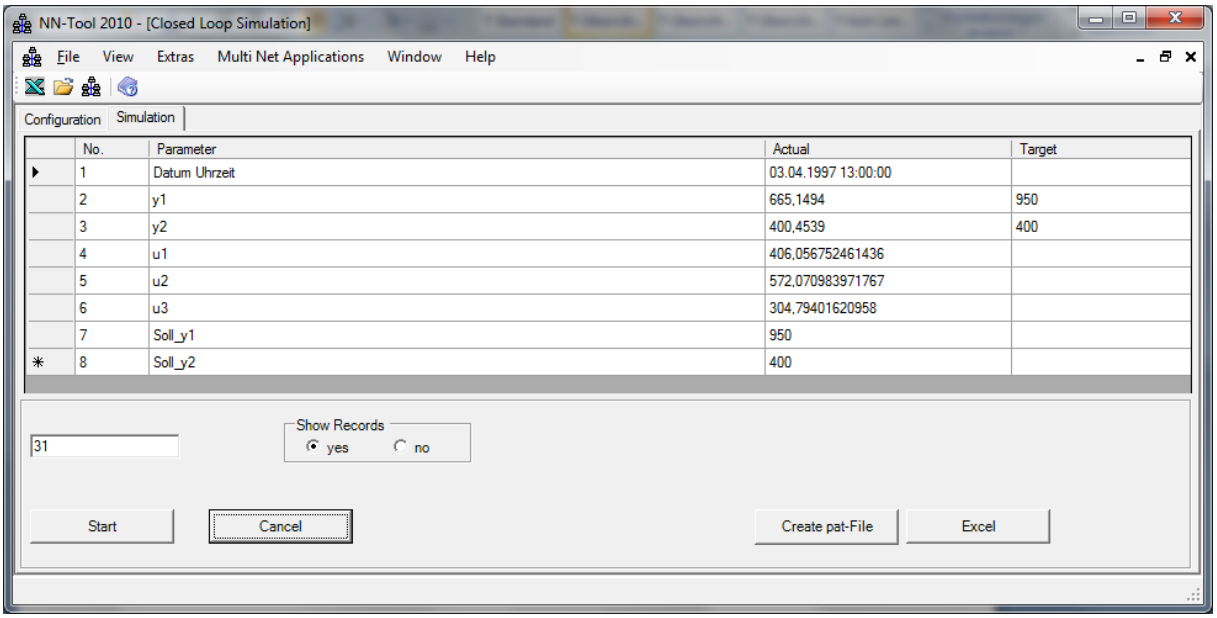
- Generally you will select **different applications** (based on different networks) for the controller and the process to study the influence of model errors ("**Model Plant Mismatch**") and the effect of the offset compensation. For compatibility, however, both networks should have been created with the same .pat-file.

- The .mna-application for the **process** must not have specified manipulated nor controlled variables.

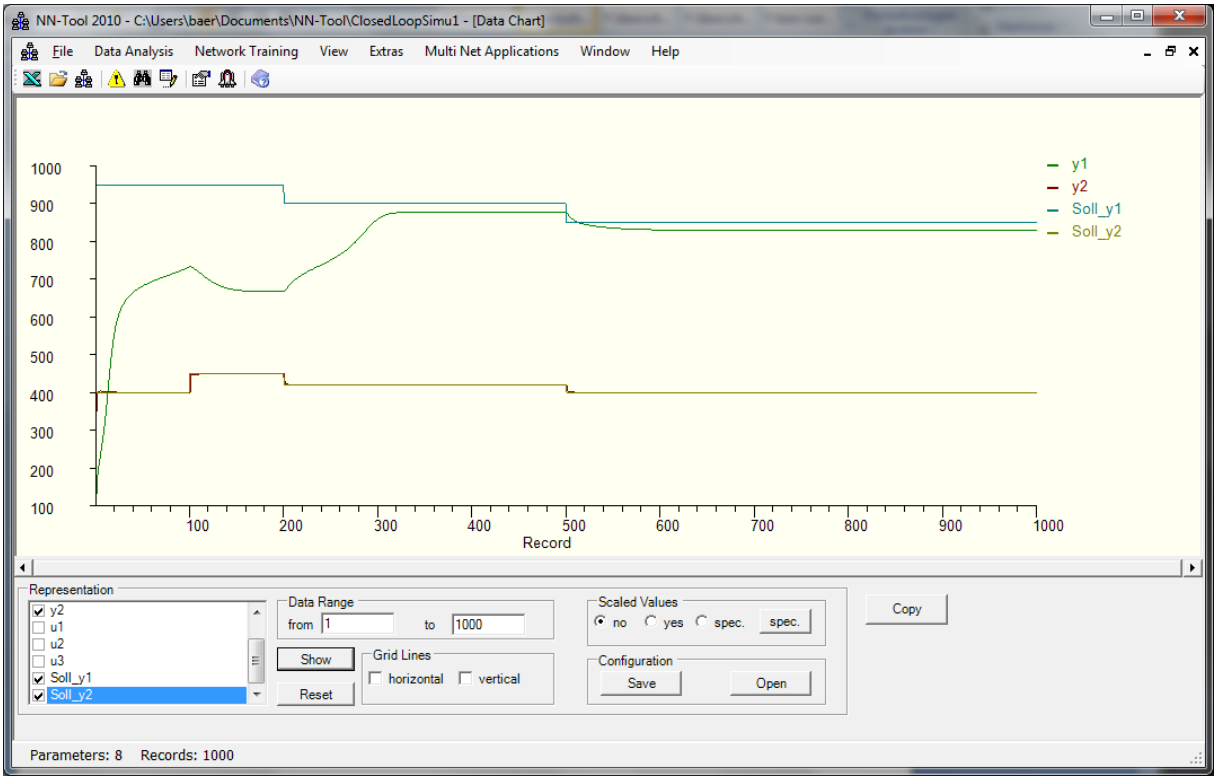
- The input file of the controller application ("Simulation Input File") is used by default as a basis for the simulation.



Save the configuration, go to the tab "Simulation" and press the button "Start".



By pressing the button "Excel" all the results can be transmitted into an Excel sheet. More efficient is the usage of the key "Create pat-file". The simulation results are stored as a .pat-file, which is then loaded and analyzed:



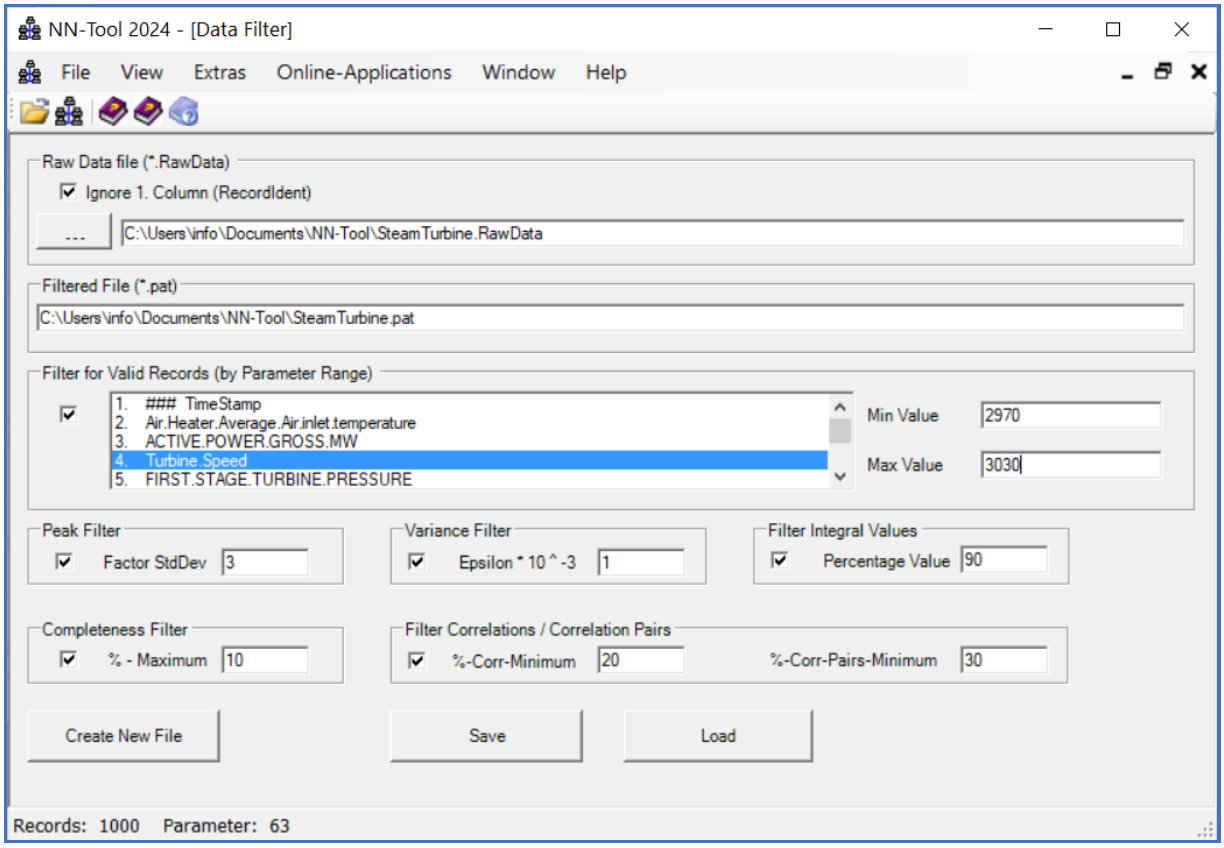
It can be seen virtually perfect control behavior, but this is not surprising, since the controller model corresponds exactly with the process. Here, moreover, no offset compensation would be required.

Vary now control and process model to study the effects of "model-plant mismatch“.

Important note: Currently Closed loop simulation is limited to models in which the inputs are exactly one time cycle ahead of the outputs, i.e. inputs at cycle “-1”, outputs at “0” (i.e. outputs without shift). If necessary, the raw data must be appropriately moved against each other in Excel.

Annex 13: Data File Filter

This component in the "**File**" main menu makes it possible to create a cleaned file from a raw data file that is heavily contaminated with measurement, sensor and input errors. The raw data file must correspond to the format of a .pat file, but must have the file extension **".RawData**". The cleaned file is then of type **".pat**" and can be processed further immediately.



The component performs the following cleaning and filtering steps in the order specified. The first data column, which in many cases contains a record identifier (e.g. date/time), can be excluded from the filtering:

- **All non-numeric entries** are initially removed, i.e. replaced with an empty value.

- **Filter for Valid Records:** Eliminates records. Using a selected parameter (here parameter "Turbine Speed"), impermissible operating or process states can be completely removed. This means that every data set in which the selected parameter assumes a value outside the range defined by the minimum and maximum values will be replaced by nothing but empty values in all parameters. This also applies if the selected parameter does not exist in the current data record.

- **Peak Filter:** Eliminates **outlier values**. Any data value that deviates by more than x standard deviations from the mean of the given parameter is replaced (removed) with a blank value. The factor x can be specified. Recommended range 3 – 4.

- **Variance Filter:** Eliminates entire **parameters (data columns**). Any parameter that has too little variance in its values is completely removed. The criterion is that the standard deviation must be at least epsilon * absolute value of the mean. A value for epsilon can be specified. Recommended range 1 – 10. The value is then automatically multiplied by 10^-3. This means that a default value of 2 means that all parameters are eliminated for which the standard deviation does not even reach 0.002 of the mean (more precisely: the absolute value of the mean).

- **Filter Integral Values:** Eliminates complete **parameters (data columns**). The filter eliminates all integral variables of the present data set. A parameter is considered an integral quantity if the number of value changes in a positive direction far exceeds the number of changes in a negative direction (or vice versa). If more than x percent of all value changes that occur only point in one direction, the parameter is viewed as an integral quantity and eliminated. Recommended value for this percentage: 90.

- **Completeness filter:** Eliminates complete **parameters (data columns)**. The filter eliminates all variables in the existing data set that are not assigned values sufficiently often (i.e. have too many missing measured values). First, it is determined how many complete values the parameter that has been measured most frequently has. A parameter is eliminated if it does not even reach x percent of this value (i.e. the number of values of the most frequently measured parameter). Recommended value for this percentage: 10 – 30.

- **Filter Correlations / Correlation Pairs**: Eliminates complete **parameters (data columns).** The filter eliminates all variables in the existing data set that either have a too low correlation with all other parameters (exact amount of the correlation coefficient) or for which there are too few data sets in common with other parameters to calculate the correlation ("correlation pairs"). The first criterion **"% Corr Minimum**" requires that the magnitude of the correlation coefficient with at least one other parameter has a value of at least x percent. Recommended value 10 – 30. The second criterion **"%-Corr-Pairs-Minimum**" requires that there are at least x percent times as many correlation pairs as the parameter that has the most correlation pairs. Recommended value 20 – 50.

- **Final Filter:** Always executed. Eliminates entire **parameters (data columns).** The filter eliminates all variables in the existing data set that no longer have any measured values or only always have the same measured value.

Hints:
- Only the activated filters are carried out (with the exception of the Final Filter). Each filter acts on the values left by the previous filter operations. The filtering operations are performed in the order specified above.
- The functionality is currently limited to applications with purely numerical data.
- Filtering is started by pressing the "**Create New File**" button. After completion, NN-Tool opens the cleaned .pat file.

Attention: In the newly created and opened .pat file, identifiers for the first column must be set again (manually)!

Annex 14: Use of neural networks in Excel / NN-Tool Excel Add-In

You can use the "**NN-Tool Excel Add-In**" (additional component available separately) to insert neural networks that you have created with NN-Tool into Excel folders. The networks then behave like a standard Excel function (e.g. the sum function).
More Information on demand.

Annex 15: Principal Component Analysis / PCATool

In certain applications, it may happen that the input parameters and / or the output parameters are not independent of each other, but that relations exist between them (equations). These equations reduce the number of the so-called degrees of freedom of the system. If a problem has 10 input variables among which 7 equations are established, the equations reduce the number of degrees of freedom to 3, i.e. the problem has effectively only 3 dimensions. If these equations are linear, the referred degrees of freedom can be determined with the principal component analysis.

PCATool is an independently running application for principal component analysis (Principal Component Analysis - PCA), which has been developed at Bayer AG, Leverkusen, and can also be purchased. Since PCATool uses the same data structures as NN-Tool, it provides for a very useful addition to NN-Tool for certain applications („Multi Net Applications"- MNA). An example can be found in the following documentation on Multi Net Applications. PCATool can also be used completely independent of NN-Tool. For details, please refer to the PCATool documentation.

Annex 16: Multi Net Applications - MNA

Multi Net Applications ("MNA") are not used to describe a single neural network, but an interconnection of multiple networks (or so-called user modules, see annex 14). **MNA is designed for the experienced user.** During the NN-Tool setup a MNA example is also supplied. These are the files **MNA_Demo_PCA.pat** and **MNA_Validation.pat** in the folder "My Documents\NN-Tool". In the following the procedure is explained by this example.

MNA is based on completed NN-Tool networks. For the example application the networks are to create first of all. Please load the file **MNA_Demo_PCA.pat**.

Hint: The file is using decimal comma. If you are using decimal point, please use an editor to replace all commas "," in this file by dots ".".

The file contains 200 records of 10 input variables x1,..., x10, followed by 9 output parameters y1 ,..., y9. It turns out that both the input variables as well as the outputs are not independent among themselves. For this reason, with PCATool a principal component analysis for the inputs (PCA_X.1 to PCA_X.10) as well as for the outputs (PCA_Y.1 to PCA_Y.9) has been made and the corresponding values have been added to the original .pat-file. Moreover, only the first 3 principal components are of importance. Because of the large measurement errors of the records we now carry out the following procedure:

1. Providing a network that maps the 10 X-values on the first 3 X-principal components. Train a network with x1 ,..., x10 as input and PCA_X.1 ,..., PCA_X.3 as output, i.e. 10 input variables, 3 output variables. Use „5-fold Crossvalidation Cyclic" in the standard option with up to 8 internal nodes. Save the network under the application name „X_PCAX" (menu "file" menu option **"Save Net as**"), i.e. X is mapped on its principal components. As the following results from the automatic documentation show the principal components are easy to learn with high accuracy. That must indeed be the case, since it is only a linear relation burdened by measurement errors.

Correl. Test Set							
No.	Outputs	Correlation	% Rel. Error	Abs. Error	a0-Coefficient	a1-Coefficient	Count
1	PCA_X1	0,9999	0,2055	0,0208	-3,24E-03	1,0009	200
2	PCA_X2	0,9999	0,1503	0,0113	-2,86E-03	0,9997	200
3	PCA_X3	0,9998	0,3447	0,0163	-6,83E-04	0,9994	200

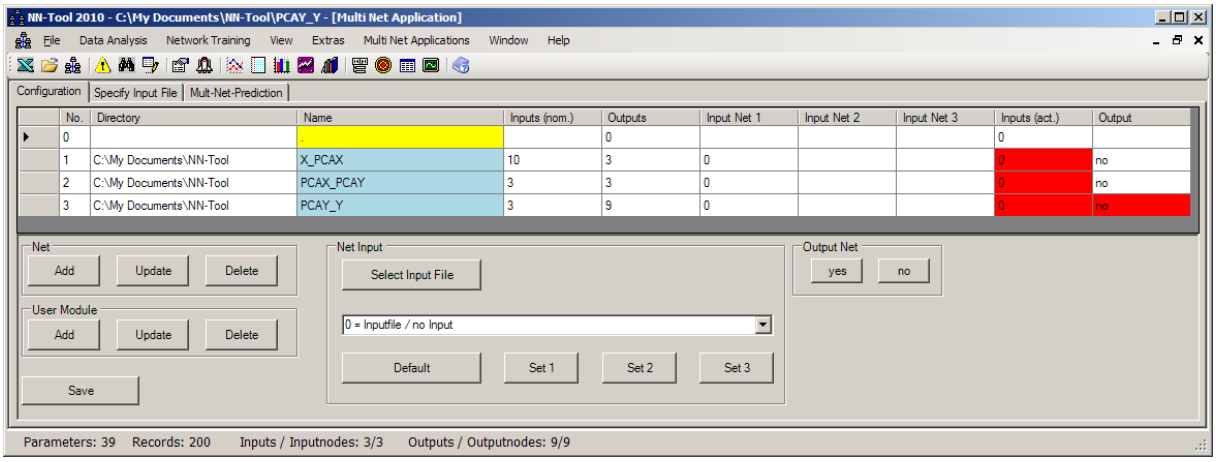
2. Train in accordance with a network using PCA_X.1 ,..., PCA_X.3 as inputs and PCA_Y.1 to PCA_Y.3 as outputs and save it under the name PCAX_PCAY.

Correl. Test Set							
No.	Outputs	Correlation	% Rel. Error	Abs. Error	a0-Coefficient	a1-Coefficient	Count
1	PCA_Y1	0,9583	3,0417	0,4733	-0,0704	0,8874	200
2	PCA_Y2	0,9353	4,855	0,3994	0,0541	0,9471	200
3	PCA_Y3	0,9547	4,2282	0,2515	-4,03E-03	0,9592	200

3. At last you create a network with PCA_Y.1 to PCA_Y.3 as inputs and y1 ,..., y9 as outputs and save it under the name PCAY_Y.

Correl. Test Set No.	Outputs	Correlation	% Rel. Error	Abs. Error	a0-Coefficient	a1-Coefficient	Count
1	y1	0,9907	1,8034	0,0548	0,019	0,9804	200
2	y2	0,9963	1,1766	0,0515	-2,66E-03	0,9926	200
3	y3	0,9946	1,4143	0,0633	-7,30E-03	0,9813	200
4	y4	0,9878	1,2801	0,0861	0,0401	0,9511	200
5	y5	0,9915	1,2201	0,066	-0,0284	0,9647	200
6	y6	0,981	2,1743	0,0861	0,0802	0,9218	200
7	y7	0,9847	2,0068	0,1413	-0,1093	0,9324	200
8	y8	0,9875	1,8647	0,1104	0,0828	0,9438	200
9	y9	0,9964	0,9863	0,0657	5,81E-03	0,9902	200

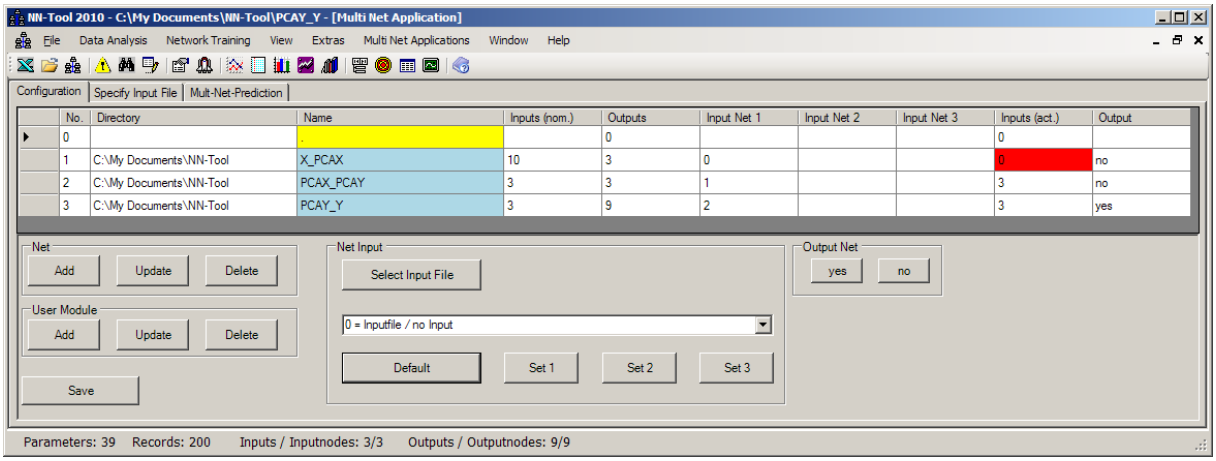
These 3 networks will now form a series connection using MNA. MNA not only allows linear connections, but any connections of networks and user modules. Each network (or user module) gets its input from the outputs of up to three upstream networks (or User Modules). So the input and output numbers of the relevant networks must fit, of course. If necessary an adaptation of the outputs of a given network can be done by switching in a suitable user module. The first network will get its inputs either on a file with input data ("Input File Prediction") or in use as an online application the data is transferred from a process control system. Open **"Multi Net Applications" / "New Multi Net Application"** and add the 3 already created networks using the „Add“ button (for "Net").



The red boxes indicate that the respective input currents (Inputs act.) in the three networks have not been specified suitably. In addition, at least one network output („Output") has to be established, usually the last.

For each network (or user module) up to 3 pre-nets can be specified (with "Set 1", "Set 2", etc.) whose outputs stream in that network as inputs. The inputs of the given network will be specified in such a way that first, the outputs of the first upstream network „Input Net 1“ will be the first source of input, then the outputs of the second upstream network, etc. Obviously, the sum of the inputs of the upstream networks (" inputs (act.) ") must match with the number of inputs of the given network ("Inputs (nom.)"). For the first network an input file must be assigned instead, and then the parameters of the file must be assigned to the inputs of the first network. For a series connection (as here) in which each network gets its inputs only from the outputs of the previous network, you can set the interconnections by pressing the button

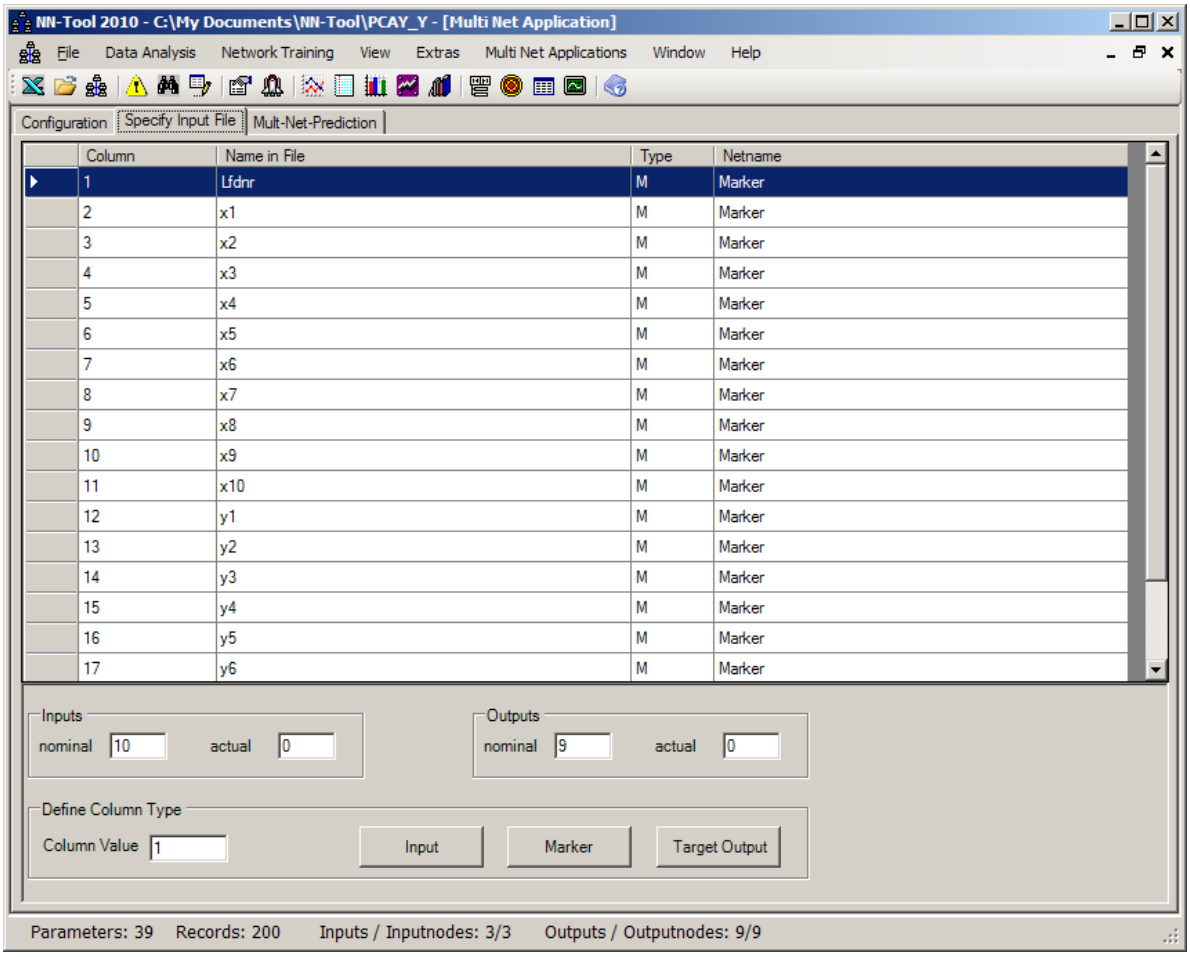
"Default".



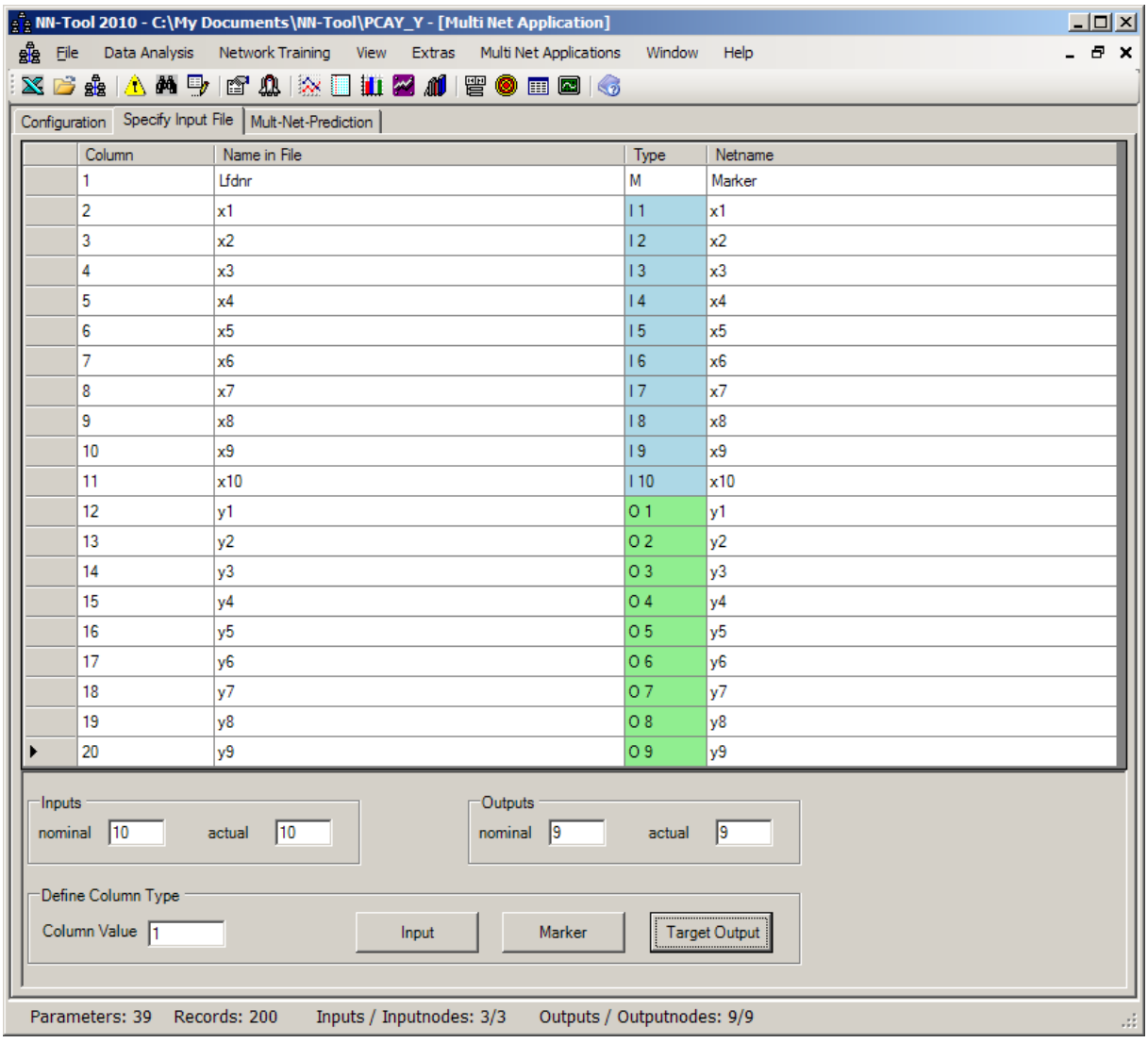
So net 2 "PCAX_PCAY" gets its input from network 1 ("X_PCAX"), etc. Now specify the file **„MNA_Validation.pat“** as input file by pressing "Select Input File".

Hint: The file is using decimal comma. If you are using decimal point, please use an editor to replace all commas “,” in this file by dots “.”.

It follows the following window:

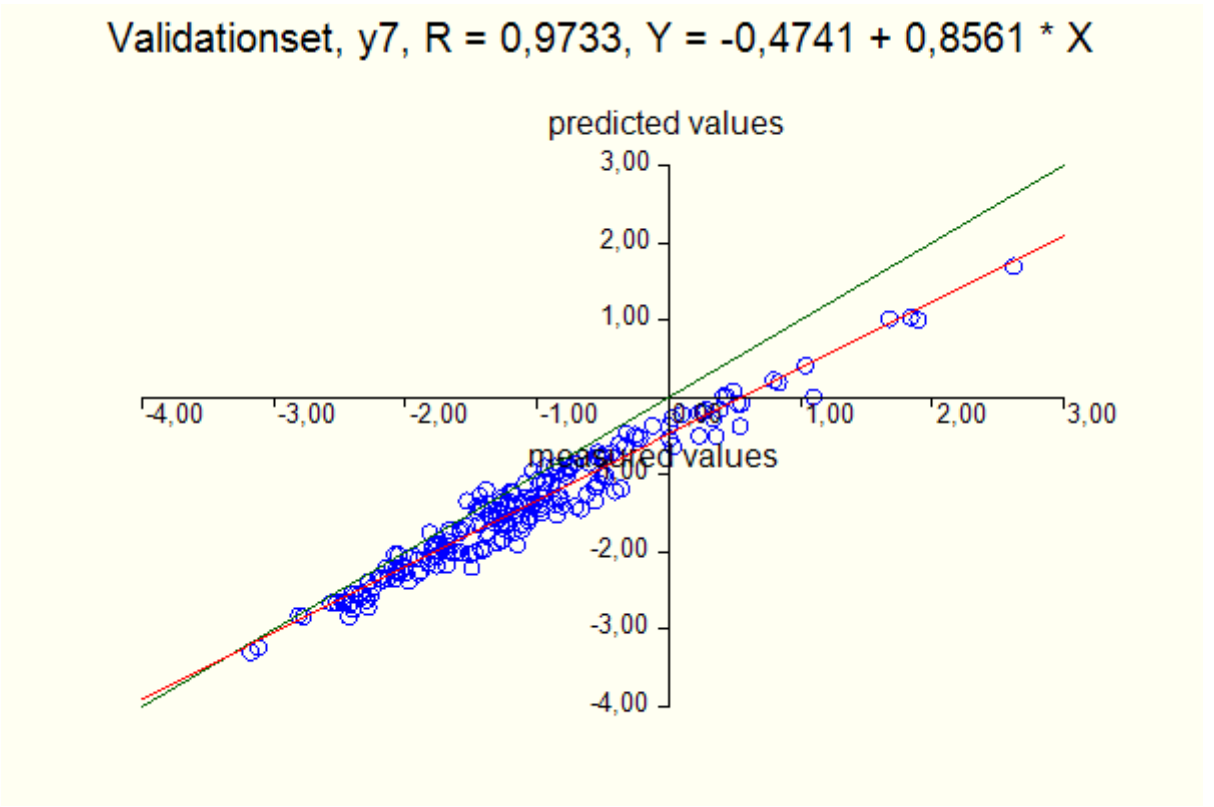


This approach is more flexible than in the „Prediction Validationset“ option, because the selected file must not have a fixed prescribed structure. To do that you now have to define the appropriate columns of the file as inputs (X-values) and possibly as target outputs (Y-values). The specification of targets may be waived if new records are just to be predicted. At the moment targets can only be established for the network with the highest number. Upon request, however, the outputs of the other networks can be predicted also and transferred to an output file or to Excel.
Define the assignment of the columns of the input file to the inputs and outputs of the multi net application as follows:

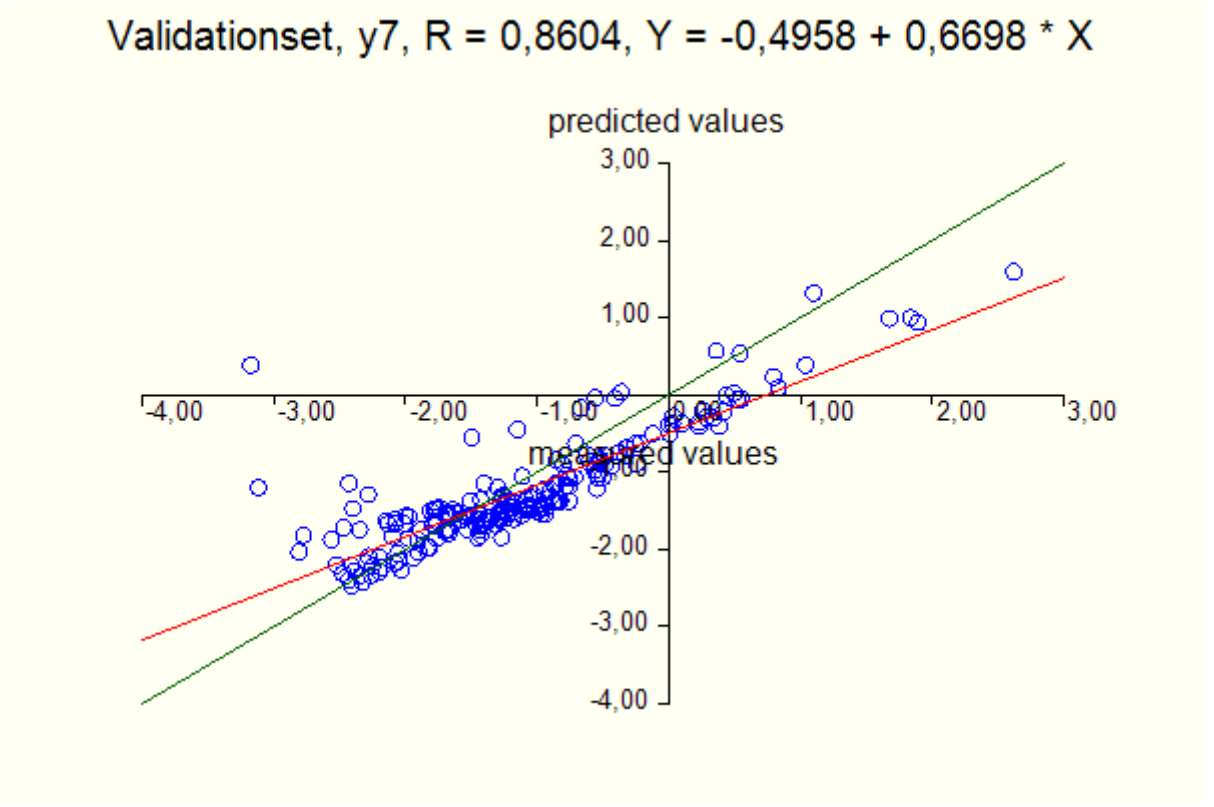


Go to the „Configuration“ tab and then save the complete multi-network application under the name "X_PCAX_PCAY_Y.mna" using the "Save" button.

On the tab page „Multi-Net-Prediction“, you can now get a prediction on the input file. This shows that the results are significantly better than in the direct prediction of y1, .., y9 as functions of x1, .., x10. For the parameter y7 the multi-network application delivers the following chart:



For comparison, the direct prediction on the file **MNA_Validation.pat** gives:



In addition to use for prediction purposes a completed multi-network application can be used especially in on-line applications (see Annex 11: Online Applications).

Note: If the following error appears at the start of the multi-network prediction:

simply rename the file in the "Print to File" Box.

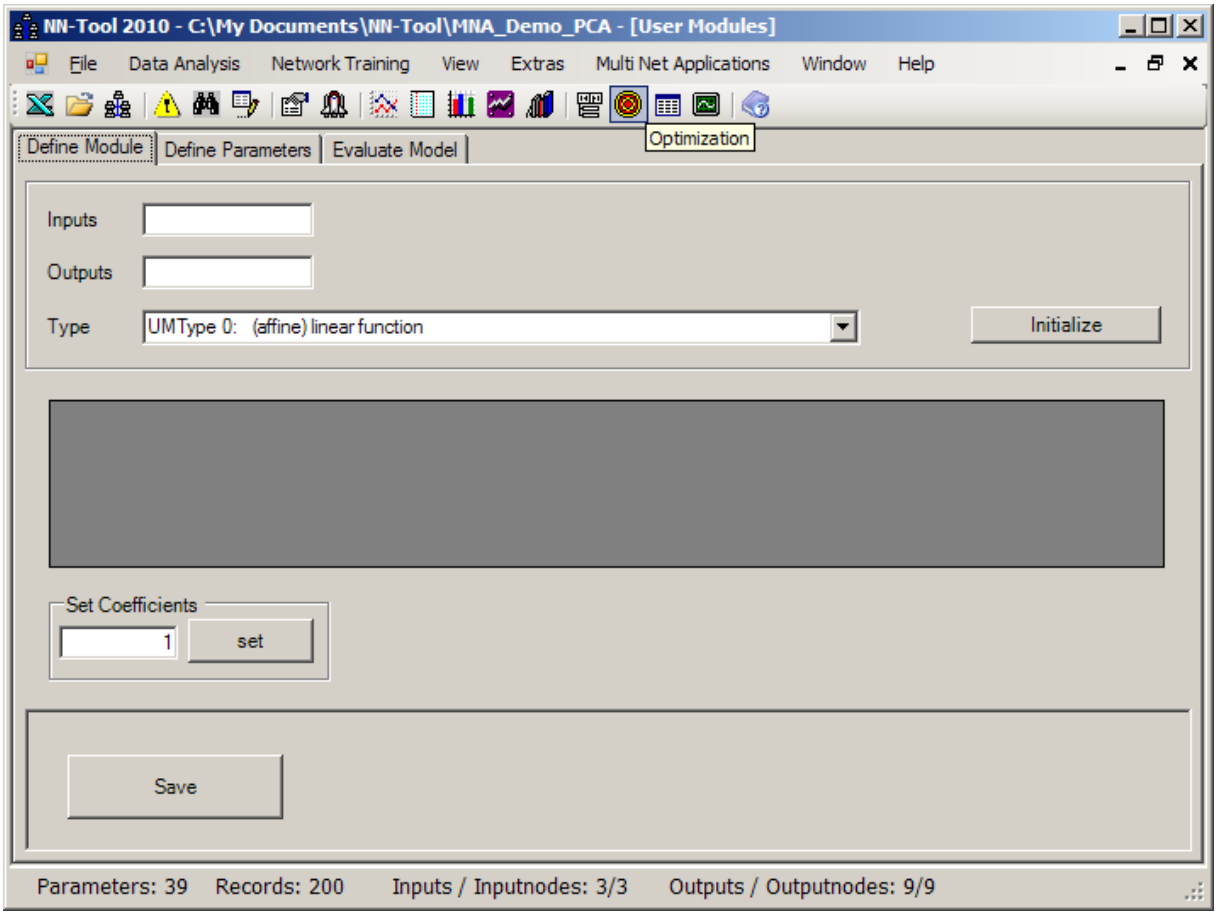
Annex 17: User Modules

The exclusive use of NN-Tool networks in multi-network applications is somewhat impractical. Thus, in some applications an additional network must be created, only to provide certain parameters. Also each time a new network is required if a subsequent net is linked only to a subset of the output parameters (or in a different order) of the previous net. User Modules now solve these problems, in which they provide the needed functionality. This is explained below on the basis of the last example of Annex 16. Please create the example applications of the previous annex first.

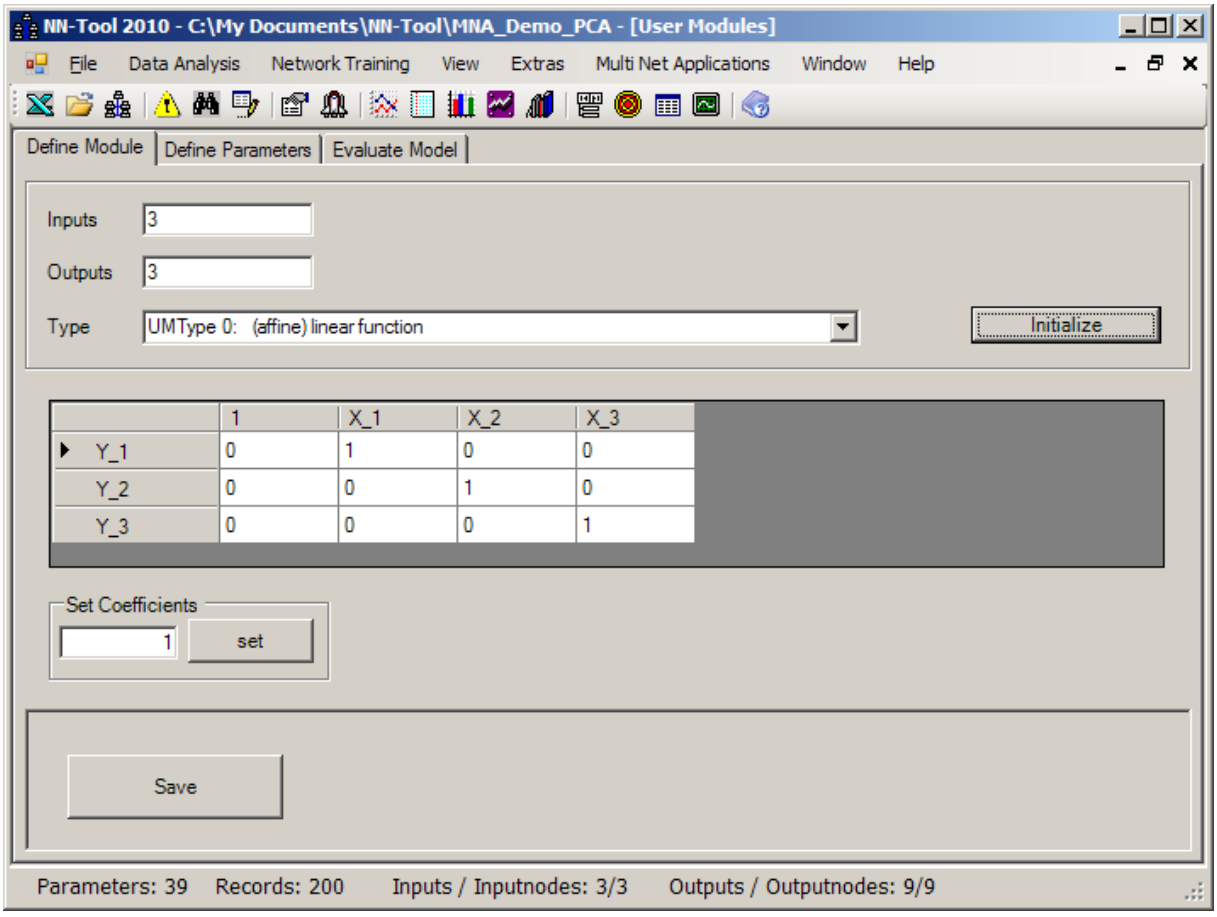
In the example of Annex 16 a multi-network application has been created on the basis of the X-values, then on the principal components of the X-and Y-values, and finally, the Y values have been predicted. We know, however, that the X-values are not independent of each other. In particular, they can not be arbitrarily varied in input files that have been created for prediction purposes. By contrast, the 3 X-principal components are independent. So it makes much more sense, to build up the multi-network application on the basis of the 3 X-principal components and to predict the resulting Y-values **and also the appropriate X-values**. For the last point a network PCAX_X is still needed, i.e. PCA_X.1 ,..., PCA_X.3 as input and x1 ,..., x10 as output. Create the network initially.

Also, the entire application needs a component PCAX_PCAX (i.e. PCA_X.1 ,..., PCA_X.3 both as inputs as well as outputs) as an input source. The reason is that currently only one network or user module can access on the input file. This first network / module can then provide all the required parameters, so that the downstream networks / modules can access them.

This functionality shall be provided now by a user module. Open the application **"New User Module"** within the main menu „Multi Net Applications":



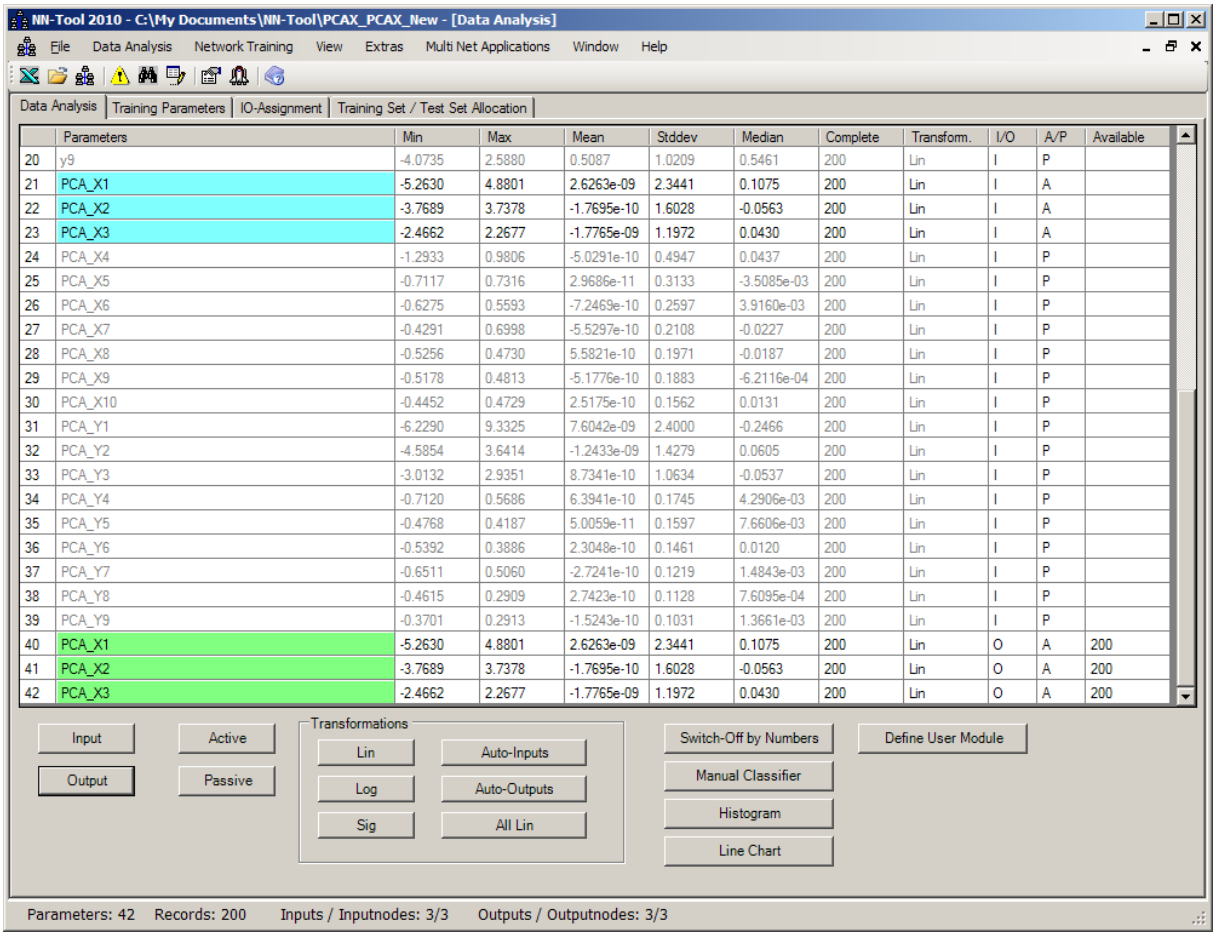
User Modules are either (affine) linear functions of any number of inputs to any number of outputs (UMType 0) or one of 19 predefined types (e.g. UMType 12 corresponds to "Sinus(x)"). To define a user module, select one of the types, specify for type 0 the desired number of inputs and outputs, and press "Initialize". Our desired functionality requires 3 inputs and outputs:



The coefficients of the (affine) linear function of 3 inputs (X_1 called up X_3) on three outputs Y_1 to Y_3 are displayed. The column "1" is used to specify constants. I.e. the first line is:

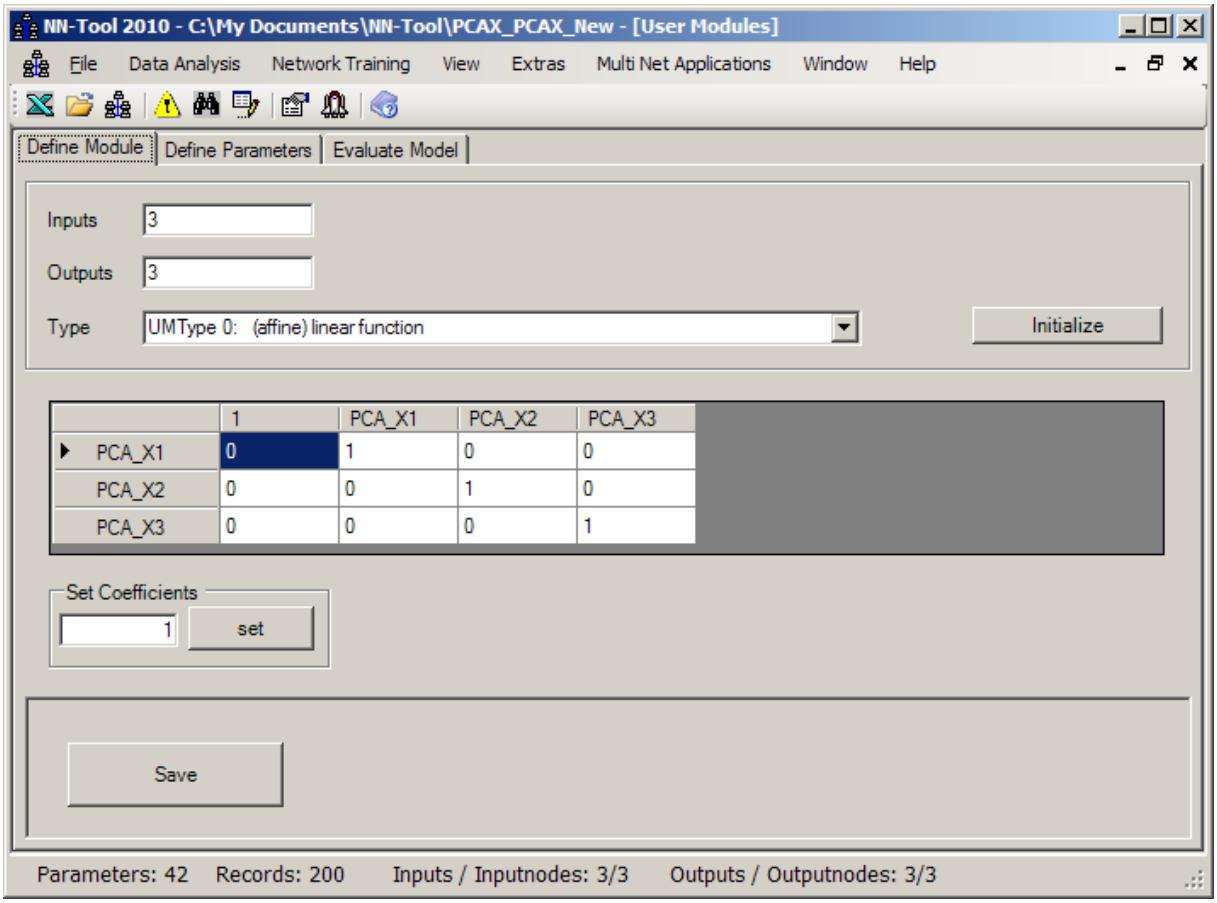
$$Y1 = 0 \cdot 1 + 1 \cdot x1 + 0 \cdot x2 + 0 \cdot x3$$

By default, all coefficients = 0 with the exception of the diagonal coefficients (= 1). With the button "set" the coefficients can be set arbitrarily. Then the model can be tested on tab page „Evaluate Model“. Since we just want to make available the values, we do not need further adjustments, and could save the model directly. The disadvantage is that the parameter have only default names and no additional information exist on acceptable minimum and maximum values (by default are 0 or 1). In order to avoid this disadvantage, a user module (from UMType 0) will be created directly from the NN-Tool data analysis window. For this we need a file in which the 3 X-principal components appear each in 2 columns. First open the already created network "PCAX_X" and save it additionally under the name PCAX_PCAX. Now open PCAX_PCAX.pat. In the window data sheet, we now need to duplicate the 3 columns for PCA_X1 to PCA_X3. Click in these columns one after the other and press the button "**Duplicate Column**“. These columns will be added. Then press „**Save Changes**“ and save the file under the name **"PCAX_PCAX_New"**. Then carry out the data analysis, set all the parameters passive except the columns for PCA_X1 to PCA_X3, each duplicate. Set the first occurrence as input, the second occurrence as output:



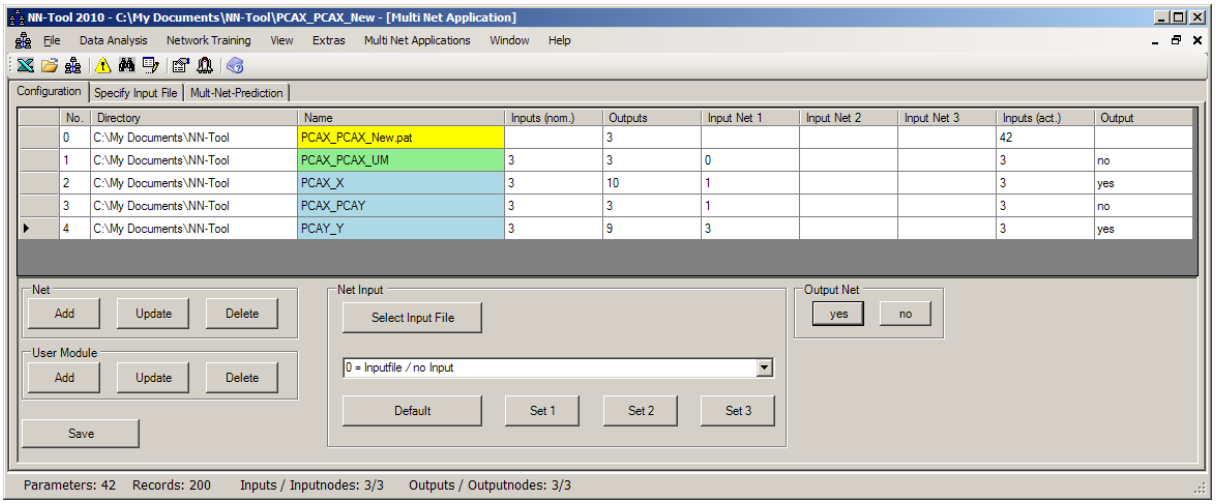
	Parameters	Min	Max	Mean	Stddev	Median	Complete	Transform.	I/O	A/P	Available
20	y9	-4.0735	2.5880	0.5087	1.0209	0.5461	200	Lin	I	P	
21	PCA_X1	-5.2630	4.8801	2.6263e-09	2.3441	0.1075	200	Lin	I	A	
22	PCA_X2	-3.7689	3.7378	-1.7695e-10	1.6028	-0.0563	200	Lin	I	A	
23	PCA_X3	-2.4662	2.2677	-1.7765e-09	1.1972	0.0430	200	Lin	I	A	
24	PCA_X4	-1.2933	0.9806	-5.0291e-10	0.4947	0.0437	200	Lin	I	P	
25	PCA_X5	-0.7117	0.7316	2.9686e-11	0.3133	-3.5085e-03	200	Lin	I	P	
26	PCA_X6	-0.6275	0.5593	-7.2469e-10	0.2597	3.9160e-03	200	Lin	I	P	
27	PCA_X7	-0.4291	0.6998	-5.5297e-10	0.2108	-0.0227	200	Lin	I	P	
28	PCA_X8	-0.5256	0.4730	5.5821e-10	0.1971	-0.0187	200	Lin	I	P	
29	PCA_X9	-0.5178	0.4813	-5.1776e-10	0.1883	-6.2116e-04	200	Lin	I	P	
30	PCA_X10	-0.4452	0.4729	2.5175e-10	0.1562	0.0131	200	Lin	I	P	
31	PCA_Y1	-6.2290	9.3325	7.6042e-09	2.4000	-0.2466	200	Lin	I	P	
32	PCA_Y2	-4.5854	3.6414	-1.2433e-09	1.4279	0.0605	200	Lin	I	P	
33	PCA_Y3	-3.0132	2.9351	8.7341e-10	1.0634	-0.0537	200	Lin	I	P	
34	PCA_Y4	-0.7120	0.5686	6.3941e-10	0.1745	4.2906e-03	200	Lin	I	P	
35	PCA_Y5	-0.4768	0.4187	5.0059e-11	0.1597	7.6606e-03	200	Lin	I	P	
36	PCA_Y6	-0.5392	0.3886	2.3048e-10	0.1461	0.0120	200	Lin	I	P	
37	PCA_Y7	-0.6511	0.5060	-2.7241e-10	0.1219	1.4843e-03	200	Lin	I	P	
38	PCA_Y8	-0.4615	0.2909	2.7423e-10	0.1128	7.6095e-04	200	Lin	I	P	
39	PCA_Y9	-0.3701	0.2913	-1.5243e-10	0.1031	1.3661e-03	200	Lin	I	P	
40	PCA_X1	-5.2630	4.8801	2.6263e-09	2.3441	0.1075	200	Lin	O	A	200
41	PCA_X2	-3.7689	3.7378	-1.7695e-10	1.6028	-0.0563	200	Lin	O	A	200
42	PCA_X3	-2.4662	2.2677	-1.7765e-09	1.1972	0.0430	200	Lin	O	A	200

and now press the button **"Define User Module"**



Save the module under the name "**PCAX_PCAX_UM**".

Now define the following multi-network application with this module:



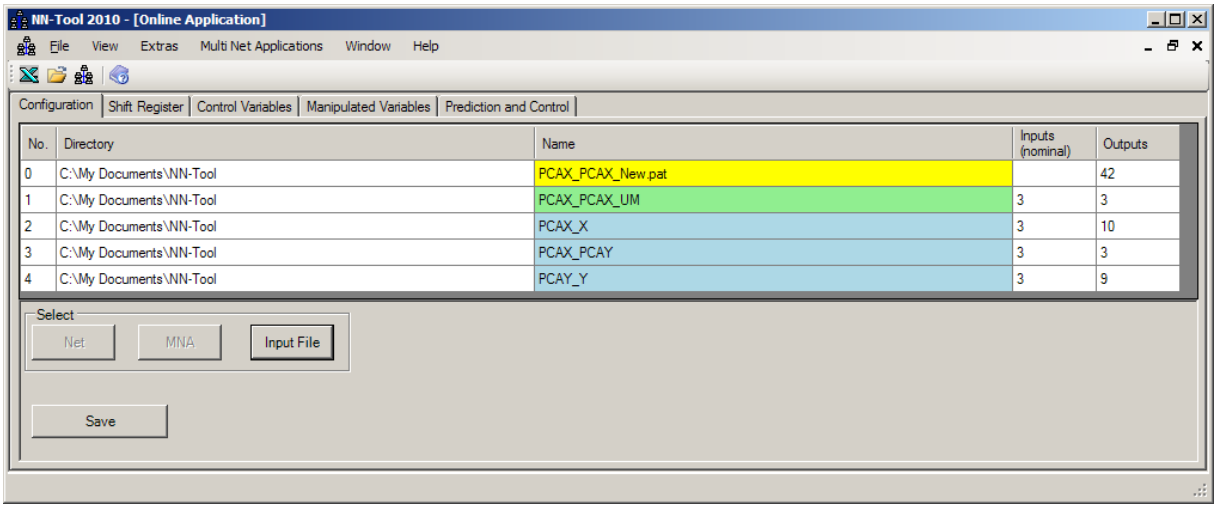
Save the entire application under the name "UM_PCAX_X_PCAY_Y".

The entire configuration can now be used to predict both the X-values as well as the Y-values on the basis of the independent variables PCA_X1 to PCA_X3 which can be read from suitable input file and, for example, can transfer the predictions to Excel or to an output file. Note that for this purpose, the network "PCAX_X" has been set as "Output".

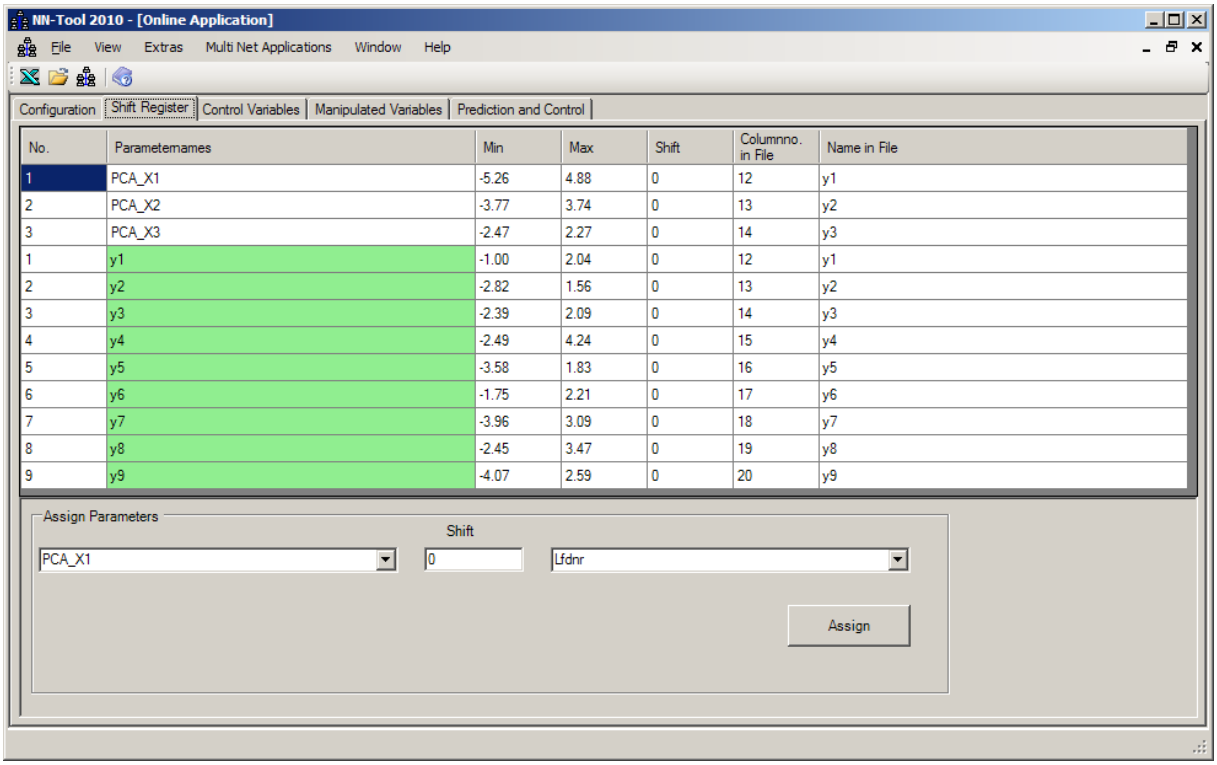
Moreover, the remaining user module types enable through suitable connections arbitrary transformations of the outputs of upstream networks / user modules and to deliver that transformed values to downstream networks / user modules. In this sense, the current restriction to a maximum of 3 upstream networks (or to just one network / user module with access on the input file) is not really a restriction. Overall arbitrary networks of interconnected networks / user modules can be defined and evaluated.

In the following we will now generate a **simulation based on this multi-network application**. Restart NN-Tool, select "**New Online Application**", tab "**Configuration**", press the button "MNA" and select the Multi-network application "UM_PCAX_X_PCAY_Y.mna" described before under User Modules. Initially, the following notice will appear:

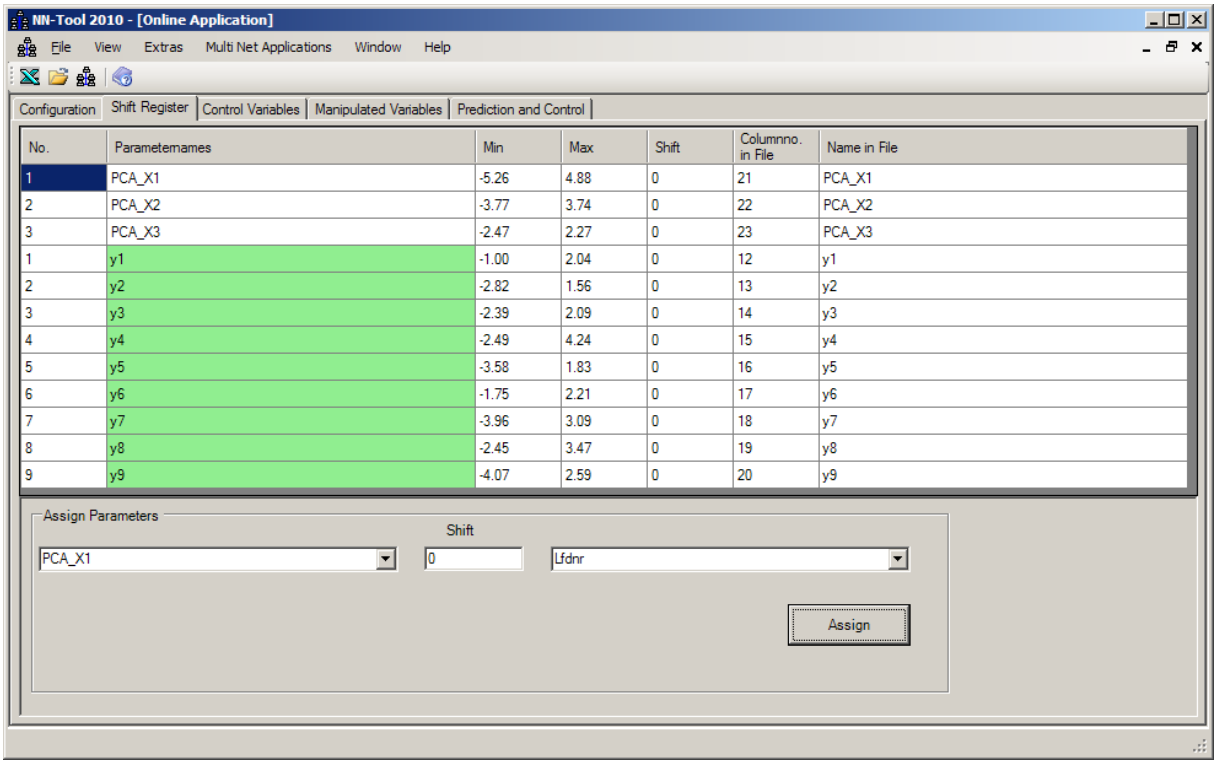
That means we still have to do some specification of the input file. Then the following window appears:



Switch to tab “Shift Register”:

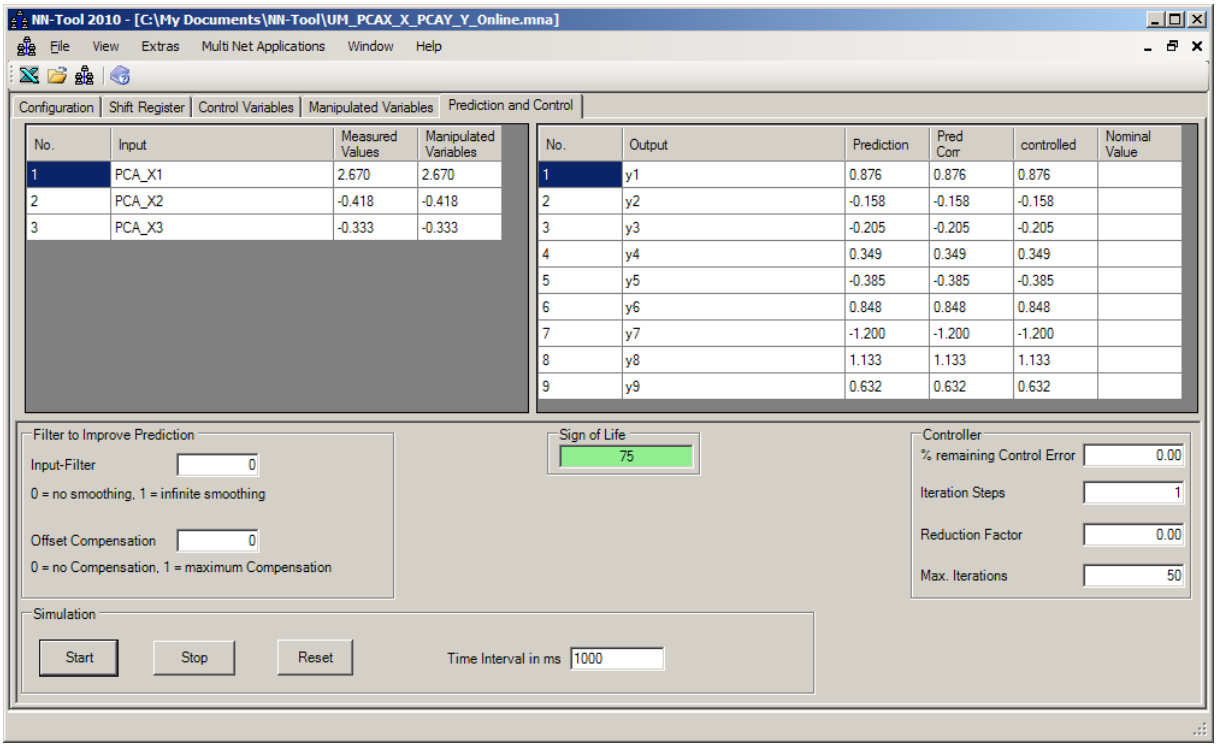


As you can see, the assignment for the outputs y1, .., y9 (green) is correct, however the inputs PCA_X1 to PCA_X3 have been assigned to the wrong columns of the file. The assignment must now be carried out individually for each input using the two list boxes and the button "Assign":



Save the configuration (name **„UM_PCAX_X_PCAY_Y_Online.mna“)**, go to "Prediction and Control" and press "Start" (possibly a message is displayed about a register length, which

was automatically adjusted):



Note: in addition to this NN-Tool component a .NET runtime library is deliverable to link NN-Tool Online Applications with external applications (in particular, process control systems, see product overview).

Annex 18: NN-Tool Network Structure

The prediction of an input vector X_{String} of length L1 of strings consists of 5 transformation steps:

1. **Coding of classifiers:** Numerical parameters are passed through to the next transformation step, but classifiers are coded in corresponding 0-1 values, i.e. input parameters for active classes receive the numeric value 1, other parameters of the classifier get the value 0. Coding is performed in order of the occurrence of class instances in the data. This transformation produces a numerical input vector X_{num} of length L2 (L2> = L1).

2. **Scaling to the interval** $[0.1,0.9]$ generates a scaled input vector X of length L3 $(L3 = L2 + 1)$. $\min, \max, \text{mean}, \text{stddev}$ are the respective values of each parameter of X_{num}. $\text{Log}(\ )$ is the natural logarithm

   a. Linear: $$x_s = 0.8\frac{x_{num} - \min}{\max - \min} + 0.1 = P_0 + P_1 x_{num}$$

   b. Log.: $$x_s = 0.8\frac{\text{Log}(x_{num}) - \text{Log}\min}{\text{Log}\max - \text{Log}\min} + 0.1 = P_0 + P_1 \text{Log}(x_{num})$$

   with $\text{Log}\max = \text{Log}(\max)$

   c. Sigmoid: $$x_s = \frac{0.8}{1 + e^{-a(x_{num} - \text{mean})}} + 0.1 = \frac{0.8}{1 + e^{(P_0 + P_1 x_{num})}} + 0.1$$

   with $a = \frac{1.6}{\text{stddev}}$

   d. In addition, a constant value 1 (= bias) is appended as the last component of the vector.

1. **The Neural network** mapping produces a scaled output vector Y of length L4:

$$Y_r = \sigma\left(\sum_{k=1}^{n} net_k w_{k,r}\right) \text{ for } r = 1,..,L4$$

$$\text{with } net_k = \sigma\left(\sum_{i=1}^{L3} x_i v_{i,k}\right) \text{ and } \sigma(x) = \frac{1}{1 + e^{-x}}$$

2. **Rescaling of outputs** generates vector Y_{num} of length L4:

   e. Linear: $$y_{num} = \frac{(y_s - 0.1)(\max - \min)}{0.8} + \min = P_0 + P_1 y_s$$

   f. Log.: $$y_{num} = \mathrm{Exp}\left(\frac{(y_s - 0.1)(\mathrm{Log}\max - \mathrm{Log}\min)}{0.8} + \mathrm{Log}\min\right) = e^{P_0 + P_1 y_s}$$

   g. Sigmoid: $$y_{num} = -\frac{1}{a}\mathrm{Log}\left(\frac{0.8}{y_s - 0.1} - 1\right) + \mathrm{mean} = P_0 + P_1 \mathrm{Log}\left(\frac{0.8}{y_s - 0.1} - 1\right)$$

   with $a = \frac{1.6}{\mathrm{stddev}}$ and limitation of y_s to the interval $[0.10653, 0.89347]$

3. **Recoding of output classifiers:** Numerical parameters are passed through, for classifiers the class name of the class with the highest rating in the corresponding output parameters of Y_{num} will be selected. Generates a string vector Y_{String} of length L5 (L5 <= L4).

Annex 19: NN-Tool Product Overview

A. NN-Tool Standard Edition

Complete development and application system for prediction, simulation and optimization with neural networks. Contains among others the following main features:

- **Data Analysis Module**: data preprocessing and statistical analysis, including appropriate modules for classifiers, for correlation calculations for the determination of optimal input parameters, for time series with time delays as well as for the treatment of incomplete records.
- **Training Module**: automatic network training including network structure optimization and crossvalidation.
- **Network Analysis**: evaluation of model accuracy with scatter plots and line charts, influence analyses, outlier module to determine inconsistent records, graphics to display the functional relationships between input and output values.
- **Application Module Control Panel**: Simulation of new situations (such as recipes, operating points) on the basis of the established network.
- **Application Module Optimizer**: used to optimize application cases (recipes, operating points, etc.) taking into account a variety of constraints on the basis of the established network.
- **Application Module Experimental Design**: used to create test plans, which are optimized to the application.
- **Documentation Module**: automatic logging of the network created including all relevant charts.
- **Batch Control**: control of NN-Tool with batch files („Macro-recorder“). Serves for the "remote control" of the learning process to allow the integration of NN-Tool in company-specific application environments (especially user databases).
- **Multi Net Module**: allows the combination of multiple networks to a total application.
- **Dynamic Simulation Module**: used to simulate time series, for example, to analyze the start-up of process plants
- **Configuration and Simulation Environment for Online Applications**: preparing NN-Tool networks for online applications.
- **Online Helpsystem**

B. NN-Tool Batch Edition

Functionality as the Standard Edition with the exception that the networks can be trained only in batch mode (i.e. ASCII control file). For the integration of NN-Tool in company-specific applications.

C. NN-Tool RunTime Edition

As Standard Edition, but without any possibility for network training (but including in particular the application modules). Used to evaluate generated neural networks for prediction and optimization purposes. Networks can not be created nor modified with this component.

D. NN-Tool Excel Add-In

Allows the use of completed neural networks within Excel workbooks.

E. NN-Tool C-library

Contains functions for loading and evaluating generated neural networks as C functions. Enables the integration of the networks in any applications, including UNIX, LINUX, etc.

F. NN-Tool Class Library for Visual Basic / .NET

Contains functions for loading and evaluating generated neural networks as VB functions. Enables the integration of the networks in any Visual Basic applications.

G. NN-Tool Control Server

Component for the realization of online applications, e.g. soft sensor applications or model based predictive control, based on NN-Tool process models (ActiveX Library).

H. Maintenance Contracts

For all the components mentioned above maintenance contracts are available. These include the elimination of emerging software errors as well as the delivery of the current versions.

Annex 20: GNU LESSER GENERAL PUBLIC LICENSE

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be

introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses

the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system,

rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and

all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> <one line to give the library's name and a brief idea of what it does.>
> Copyright (C) <year> <name of author>
>
> This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.
>
> This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.
>
> You should have received a copy of the GNU Lesser General Public

```
    License along with this library; if not, write to the Free Software
    Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-
1301  USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the library, if
necessary.  Here is a sample; alter the names:

  Yoyodyne, Inc., hereby disclaims all copyright interest in the
  library `Frob' (a library for tweaking knobs) written by James Random
Hacker.

  <signature of Ty Coon>, 1 April 1990
  Ty Coon, President of Vice

That's all there is to it!
```

Annex 21: Installation and Runtime Errors

NN-Tool menu item is not available in Excel:

If Excel is started form NN-Tool for the first time, NN-Tool tries to install in Excel a so-called add-in named "Nnt3_Englisch.xla" (not to be confused with the add-in under Annex 17). An add-in is an additional component (a "macro") in Excel, which can be used to extend the functionality of Excel. The add-In nnt3_Englisch.xla (name nnt3_Englisch name, file extension .xla for Excel Add-In) is used to convert the data of an Excel spreadsheet into a NN-Tool data file (*.pat file) in a comfortable way. Depending on the set of safety restrictions and the Excel version the automatic installation of the add-In may fail. Then, proceed as follows:

- If you have Excel 2007 (Office 2007): In this case, the menu is on the tab "Add-Ins." See there first.
- Try to install the add-in manually. Use the command "Add-In Manager" in the "Extras" menu. (Office 2007: Please see the Help for how to install add-ins). The Add-In Manager will be opened.
- With it you can browse the NN-Tool installation directory to find the add-in „Nnt3_Englisch" and install it. If you do not know in which directory you have installed NN-Tool, you can find the directory name in NN-Tool under „Extras/Options".
- If even that does not work, there is also the possibility of an earlier version of the add-in (called nnt3_old_Version.xla) to install.
- Alternatively, there is the possibility of a **direct access to an Excel-folder**: instead of a .pat-file NN-Tool initially loads an Excel application, the user selects one worksheet (with several sheets in the Excel folder) and NN-Tool creates and imports the data file (see data preprocessing).
- If all options fail (i.e. the add-in does can not be installed or run, an access to Excel is not possible), it is necessary to convert the Excel data with Excel methods itself. To do so, the Excel spreadsheet has to be saved using the "Save As" option with the file type ".csv" ("character separated values“) format. Then close Excel. If an indication of the nature of: *"File does not have the Microsoft Excel format. If ...",* then click "no", (i.e. save file in the .csv-format without saving changes).
- This creates a file of .csv format in which the data are separated using a list separator. This list separator depends (sorry!) on the Windows settings, specifically from the so-called Regional Settings. With German regional settings, it is the semicolon, in English it is the comma. A .csv-file can be imported by NN-Tool using "Load Datafile". You have to set the file type to csv. After that the above-mentioned list separator is asked.
- If your .csv file uses the semicolon (the default setting for German Windows versions), you can click the "OK" button. Otherwise, you have to specify the comma (or possibly even another list separator). Only one character may be entered. Then NN-Tool imports the data file, creates an additional .pat-file and you can continue as described in the manual.

Folder “My Documents\NN-Tool” does not exist.

If you can’t find that folder please copy the folder **“NN-Tool”** in the NN-Tool installation directory. The path of the installation directory can be found under “Extras“ / “Options“.

Training algorithm doesn’t start

Error message “3 Training terminated with failure. Net possibly incorrect!” is displayed. Cause: Installation dir or application dir or some files are write-protected (especially under WIN 7). Reinstall NN-Tool and use default installation directory.

Excel Communications is not working properly

- **Symptom**: the transfer of data to excel does not work. No Excel folder opens.
- **Cause**: A variety of Excel processes has been started in the background without an Excel application is visible.
- **WORKAROUND**: Start the Task Manager (<Strg> <Alt> <Del>), go to the tab "processes", sort by "name" and stop all processes called "excel.exe". Ignore the warning.
- **Background**: In some versions of Excel it may happen that when you quit an Excel application not all related Excel processes are always terminated.

Data are not interpreted correctly or model is very bad

Causes:

- In many such cases the fact is that the specifications for the decimal separator are not set correctly, for example, because the data come from another computer with other regional settings, or the regional settings on your computer have been modified. Open the .pat-file (if necessary .csv-file) with an editor and check whether the decimal point and the list separator used match with the regional settings on your computer. (The introduction of regional-specific settings was probably the biggest stupidity of Microsoft in their long list). Change the character by "Find and Replace" in the file. Possibly you also need to retrain the whole system.
- your data include additional characters (e.g. ">") besides numeric values and are therefore treated as classifiers by NN-Tool. Clean up the data. Hint: Excel numeric data are right in the cells; none are left (even with the wrong decimal separator).
- Your data have large measurement errors. Make one or several control experiments with the same input data and analyze the variation in the output data. Your models can not be better than your data.
- One or more important input variables are not available (not measured).

- There are only very few records available. Use the methods of crossvalidation in particular the "leave-one-out" option.

No solution found? Contact:

Email: info@baermann.de
Tel: +49 211 7489973