

Comparative study of fuzzy logic and neural network methods in modeling of simulated steady-state data

M. Järvensivu and V. Kanninen

Laboratory of Process Control, Department of Chemical Engineering, Helsinki University of Technology, Kemistintie 1, FIN-02150 Finland.

Abstract

In this paper fuzzy logic and neural network methods were used to model simulated nonlinear steady-state data. Two different cases of training and checking data sets were generated: ideal data without noise and realistic data with added noise and other nonidealities. Both techniques were fitting the ideal case data almost perfectly. The fuzzy logic and neural network models were also able to roughly predict the realistic case data.

1. Introduction

Both neural networks and fuzzy logic inference systems have been proved to have capabilities of universal approximators. Thus they can both be utilized in modeling of complicated nonlinear processes (Juditsky¹, Wang²).

Rotating disk filter is a complicated nonlinear process, where the output of the process is nonlinear function of a set of input variables (Kanninen³). Most important input variables are feed density, filter pressure, slurry level and rotating speed. Disk and cake flow resistances are also as important variables: their can be viewed either as input variables or time-varying parameters of the process. Cake resistance depends on time-varying physical and chemical properties of the feed slurry and on the process conditions (pressure, slurry level, rotating speed). Disk resistance depends on different fouling phenomena and thus on the past time history of the operation of the disk filter.

In this work, both neural network and fuzzy logic methods have been tested to model the rotating disk filter process. Feedforward neural networks trained by Levenberg-Marquard optimization (Demuth and Beale⁴) and Adaptive Neuro-Fuzzy Inference System, ANFIS, combined with subtractive clustering to initialize ANFIS (Jang and Gulley⁵) were used as tested methods.

1.1 Neural networks

Neural networks are composed of many simple element operating in parallel and the network function is determined by the connections between elements. The neuron model and the architecture of a neural network describe how a network transforms its inputs into an output and place limitations on what a particular neural network can compute. The training technique used, determine how well and fast neural network learn the system behavior.

One of the most important attributes of a layered neural network design is the architecture. The number of input nodes and output nodes are simply determined by the dimension of the input and output vectors to be approximated (Zurada⁶). The size of a hidden layer is one of the most important considerations when solving actual problems. There is not conclusive answer available for hidden layer sizing problem only general rule, “the bigger the hidden layer the better the network can approximate the function and if far too big hidden layer is used overfitting can occur”, can be applied with trial and error.

Although most activation functions, often called transfer functions, sum their input signals basically in the same manner, they are not identical in terms how they produce output response. Several different functions may be used as activation function and neural networks with same architecture have different capabilities according transfer function they employ (Boullard⁷).

Often the backpropagation training algorithms are based on gradient descent method together with some improvement techniques, as momentum or adaptive learning rate. However, more sophisticated techniques can be used for training of neural network instead of traditional backpropagation algorithm, e.g. Levenberg-Marquardt optimization method makes network training times shorter compared to simple gradient descent based backpropagation (Demuth and Beale⁴).

1.2 Fuzzy Logic

Fuzzy logic was primary designed to represent and reason linguistic form of knowledge. Design parameters like fuzzification and defuzzification methods and rule base together with membership function construction and representation describe how a fuzzy logic system operate (Reinfrank⁸, Jager⁹).

Fuzzification converts a point-wise (crisp), current value of a process state variable into a fuzzy set, in order to make it compatible with the fuzzy set representation. The function of the defuzzification is mapping the set of fuzzy outputs into a single point-wise value.

Definition of the fuzzy rulebase include the choice of the process state and control output variables and the content of the rule-antecedent and the rule-consequent. A membership function is a curve that defines how each point in the input space is mapped to a membership value.

One possibility to give fuzzy logic system learning capabilities is to combine training methods from neural network area to fuzzy logic inference systems, e.g ANFIS (Adaptive Neuro-Fuzzy Inference System) trains a fuzzy inference system with backpropagation algorithm (Jang¹⁰).

2. GENERATING DATA

2.1 Steady-state rotating disk filter model

Nonlinear steady-state rotating disk filter model was used to generate the data (Kanninen²). The basic form of the model is shown in equation (1).

$$\dot{m}_{cake,dry} = \frac{1}{2} (r_{outer}^2 - r_{inner}^2) \omega \frac{1}{\alpha} \left(-R_{disk} + \sqrt{R_{disk}^2 + \frac{4 \alpha \Delta p C_{slurry} \arccos\left(\frac{-L}{r}\right)}{\mu \omega}} \right) \quad (1)$$

In this basic form, the model is quite general and can be applied to many different type of vacuum and hyperbaric rotating disk filters.

2.2 Generating data for case 1, ideal case

Two different cases of training and checking data sets were generated. Both cases contain 100 points of training data and 100 points of checking data. Data for case 1, the ideal case, was generated by allowing the four input variables, pressure, slurry level, feed slurry density and drum speed to vary randomly inside a operating region, and then calculating the output variable, capacity of the filter, from equation (1). This means, that the data for case 1 is mathematically "ideal" in the sense, that there is no noise in the input or output variables, the process (model) structure and parameters do not change, and the output is affected only by the four known input variables.

2.3 Generating data for case 2, realistic noisy data case

When generating data for case 2, realistic noisy case, firstly white noise, which amplitude was about 10 % of the amplitude of the output variable, was added to output of the model. In addition to this, two parameters of the model, disk

and cake resistance, were added as randomly varied extra non-measurable input variables with range of $\pm 5\%$ (in case 1, those variables were constant).

This second addition affects more seriously to possibilities of achieving a good model with any modeling technique, because it means, that there is no information about two important input variables, which systematically affects to output. The purpose of this was to test how the fuzzy logic and neural network methods behave in this kind of process modeling situation.

3. MODELING OF IDEAL CASE DATA

3.1 Modeling of ideal case data using neural networks

Feedforward network utilizing Levenberg-Marquard optimization method was tested by training the network with training data and evaluating the network performance with separately checking data. Ideal case data was easy to approximate, because the data doesn't include any noisy or unmeasured disturbances and most of the networks tested was giving good results.

Many different size of network was evaluated. However, network with one hidden layer with 5 neurons was giving the best results. In general, by increasing the size of hidden layer, the training data approximation was giving better results, but checking data was showing that network was starting to overfit and error start increase. Figures 1. show the evaluation of the network performance with the data used for checking.

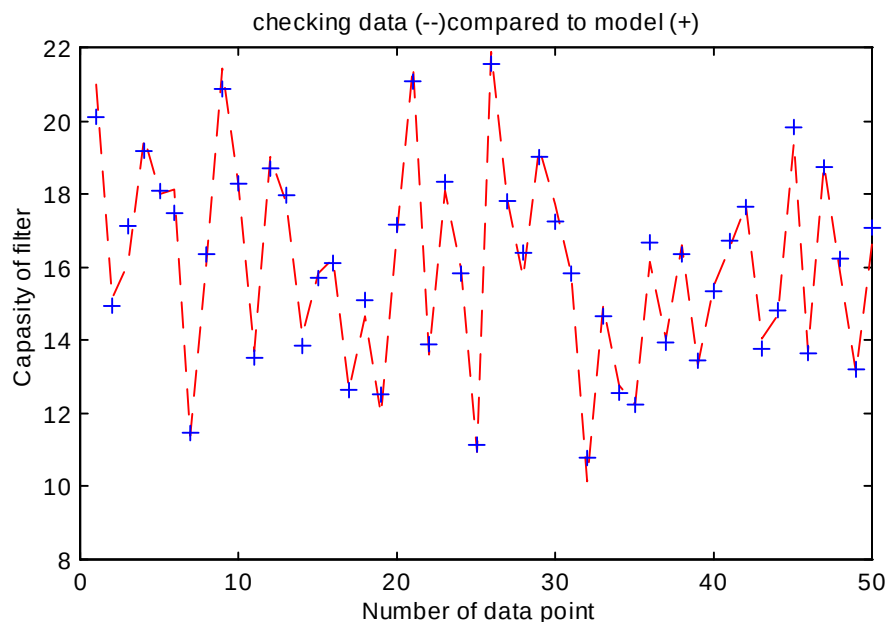


Figure 1. Comparing of checking data and the output of neural network model

3.2 Modeling of ideal case data using fuzzy logic

3.2.1 Building of Sugeno-type fuzzy model using clustering

Subtractive clustering method was used to formulate Sugeno-type fuzzy systems (Jang and Gulley⁵). The most important parameter of the function is the cluster radius, which determines how many clusters and thus fuzzy rules the function generates.

For checking data, the lowest root mean square error, 0.098, was produced by six fuzzy rules achieved by cluster radius 0.8 (the training data root mean square error was for this system was 0.054). Thus the fuzzy system with six rules was selected for further training. Figure 2 shows the root mean square error of the checking data as a function of cluster radius and figure 3 shows the same error as function of the number of the fuzzy rules.

3.2.2 Further training of fuzzy model by ANFIS

The best fuzzy system, six rules, achieved in the clustering phase was further trained using Adaptive Neuro-Fuzzy Inference System. ANFIS uses a hybrid learning algorithm to identify parameters of Sugeno-type fuzzy inference systems: it applies the least-squares method for linear parameters and backpropagation gradient descent method for nonlinear parameters. For detailed information about ANFIS, the readers are referred to Jang and Sun¹¹.

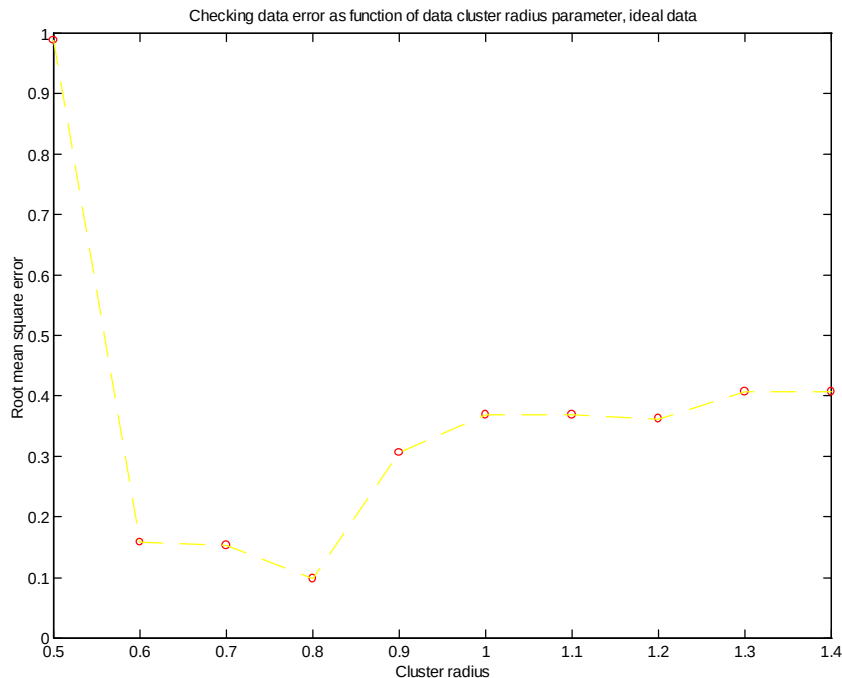


Figure 2. Error of checking data as function of cluster radius.

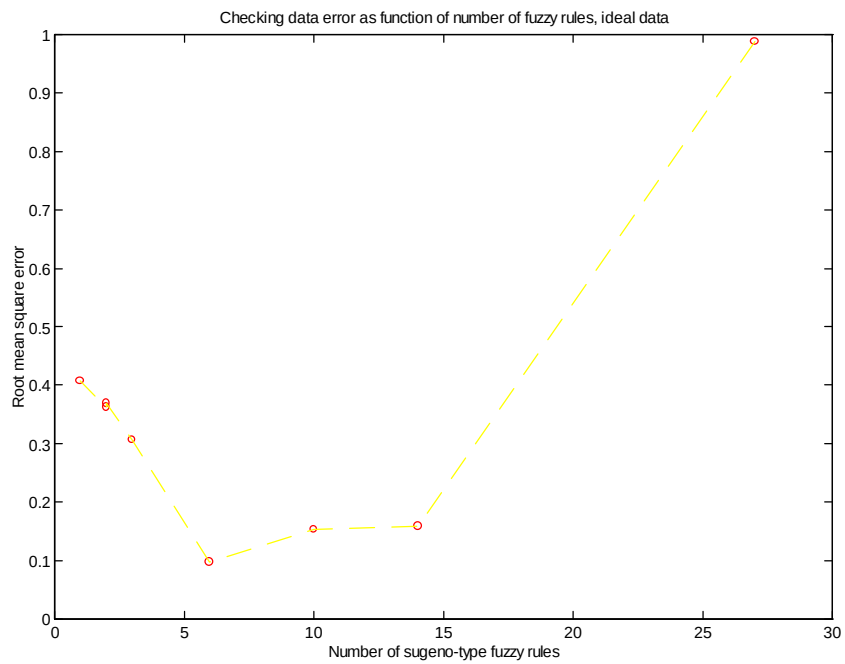


Figure 3. Error of checking data as function of the number of fuzzy rules.

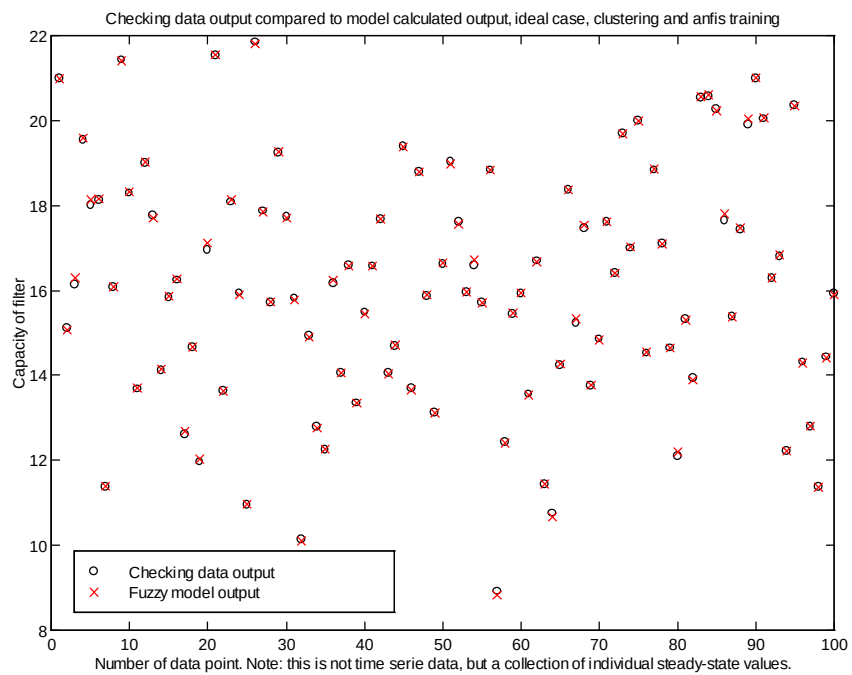


Figure 4. Comparing of checking data and the output of fuzzy model

The system with the lowest checking data error was selected the final fuzzy model for ideal data case. Figure 4. shows comparing of checking data and output of fuzzy model. This figure together with the low root mean square error value indicates, that the fuzzy model for the ideal data case can fit the data very well.

4. Modeling of realistic case data

4.1 Modeling of realistic case data using neural networks

Feedforward network utilizing Levenberg-Marquard optimization method was tested by training the network with training data and evaluating the network performance with separate checking data. Many different size of neural networks was tested and the hidden layer sizing was studied more carefully for this realistic case than for first ideal case.

The network having one hidden layer was giving best performance with checking data. Network with two hidden layer or even more were also investigated, but those were easily leading into overfitting situation where evaluation with training data was giving good results, but the performance with checking data was poor.

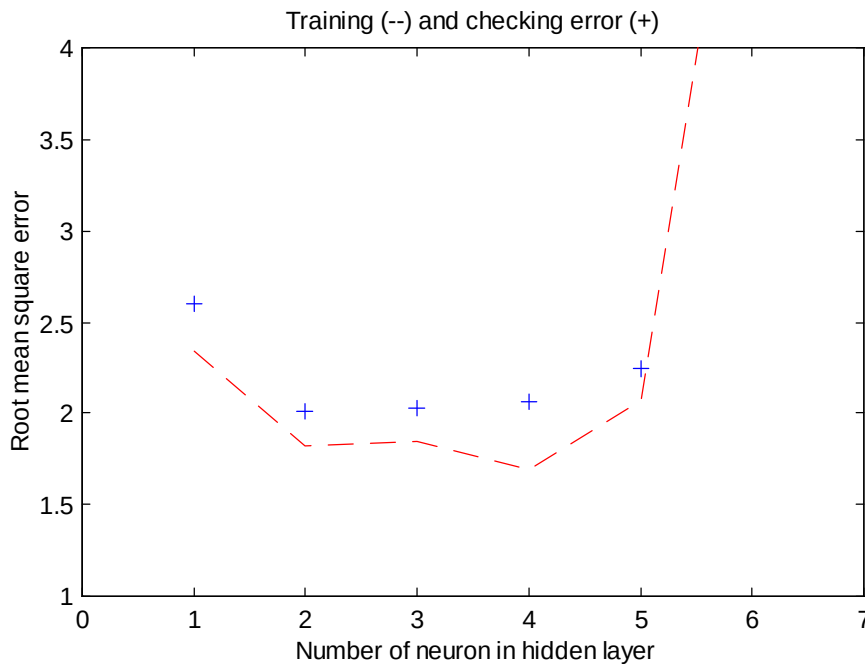


Figure 5. Hidden layer size affect on Root Mean Square Error.

The results from test runs with different number of neurons in hidden layer were quite close each other, but in general the network with three or four neurons was giving the best results. Figure 5. shows the results from one of the test runs where one hidden layer network was tested with one to six neurons and figures 6 shows evaluation of the network performance with checking data. The performance of network with realistic case data was not as good as in ideal case.

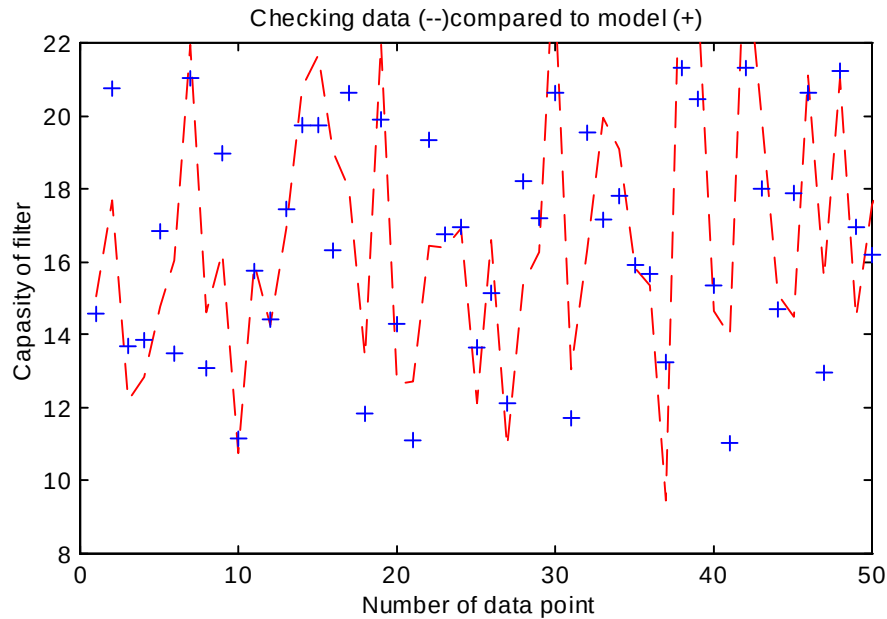


Figure 6. Checking data and model approximation

4.2 Modeling of realistic case data using fuzzy logic

4.2.1 Building of Sugeno-type fuzzy model using clustering

Figures 7 and 8 show the error of training data and the error of checking data as function of the number of the fuzzy rules. From these figures it can be seen, that as the number of fuzzy rules is increased the training data error is all the time decreased. However, for checking data the lowest root mean square error, 1.928, is produced with only one Sugeno-type fuzzy rule. This is not a real fuzzy system any more, but just a linear polynomial model, which parameters have been estimated by normal least-squares method.

System with only one Sugeno-type fuzzy rule cannot be trained further by ANFIS method (no parameters to further optimize). Because of this, a fuzzy system with 2 rules have been selected for further training, and after this, the checking data errors of one and two rules systems are compared to each other.

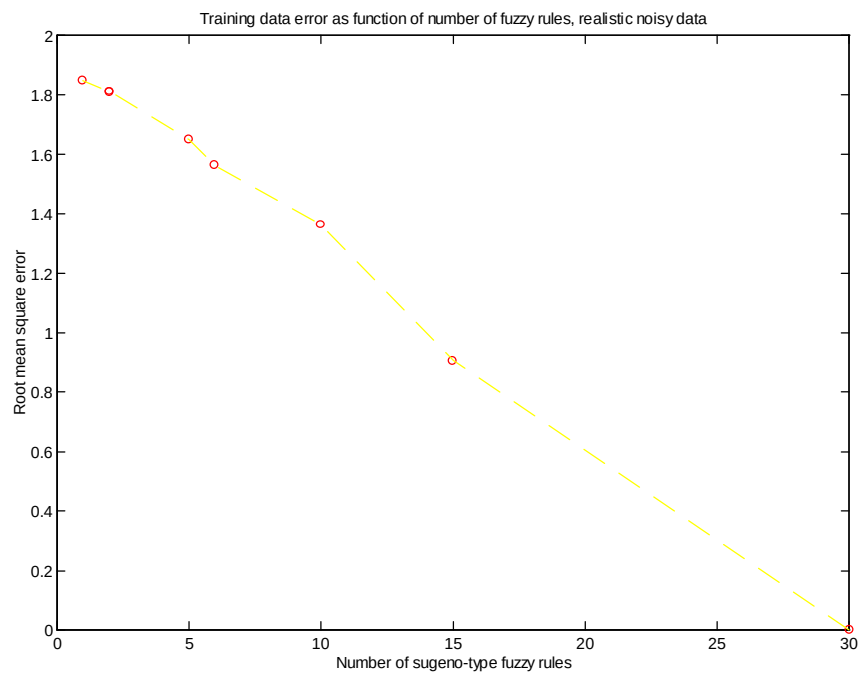


Figure. 7. Error of training data as function of the number of fuzzy rules.

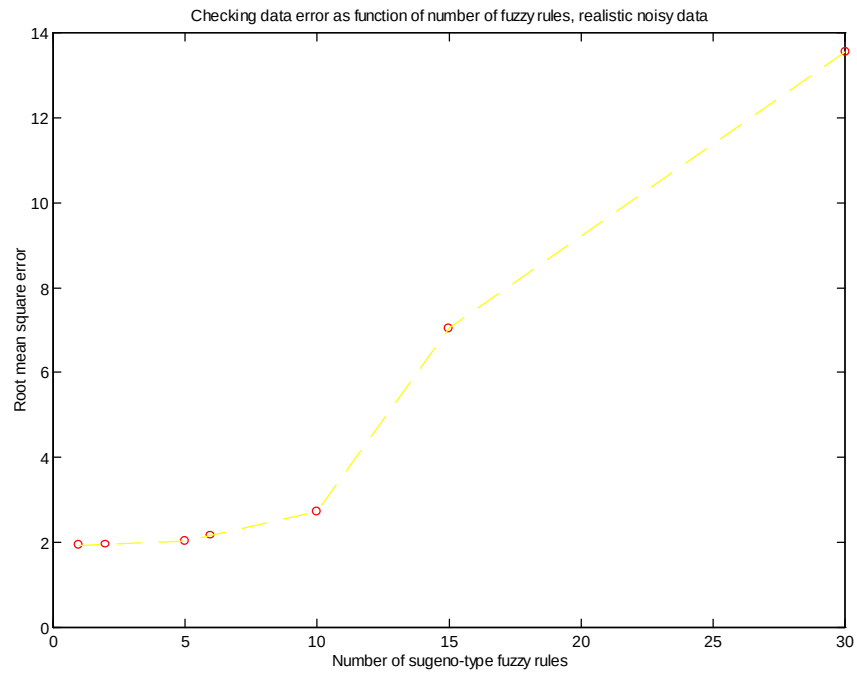


Figure 8. Error of checking data as function of the number of fuzzy rules.

4.2.2 Further training of fuzzy model by ANFIS

Fuzzy system with two rules was further trained using the ANFIS method similar way as was explained in section 3.2.2. Figure 9 shows the development of training and checking data errors during the learning. The system with the lowest checking data error was selected to be the best two rule system.

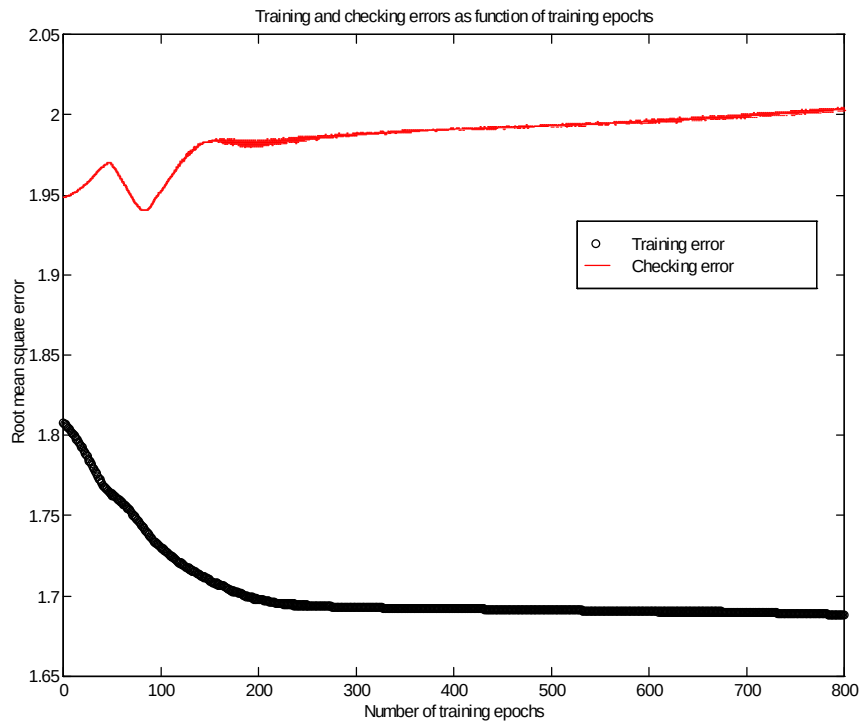


Figure 9. Training and checking data errors during the ANFIS learning

4.2.3 Comparing of model calculations and data

Because the root mean square error for checking data of one fuzzy rule model (linear model), 1.928, was lower than the checking data error for 2 rule ANFIS trained model, 1.940, the one rule model i.e. standard linear polynomial model was selected as the final model for the realistic data case. Figure 10. shows comparing of checking data and output of the model. The model for the realistic case data can only roughly forecast the behavior of the data. The reason for this is, that important information of two input variables, which are difficult to measure in practice, were not used on modeling.

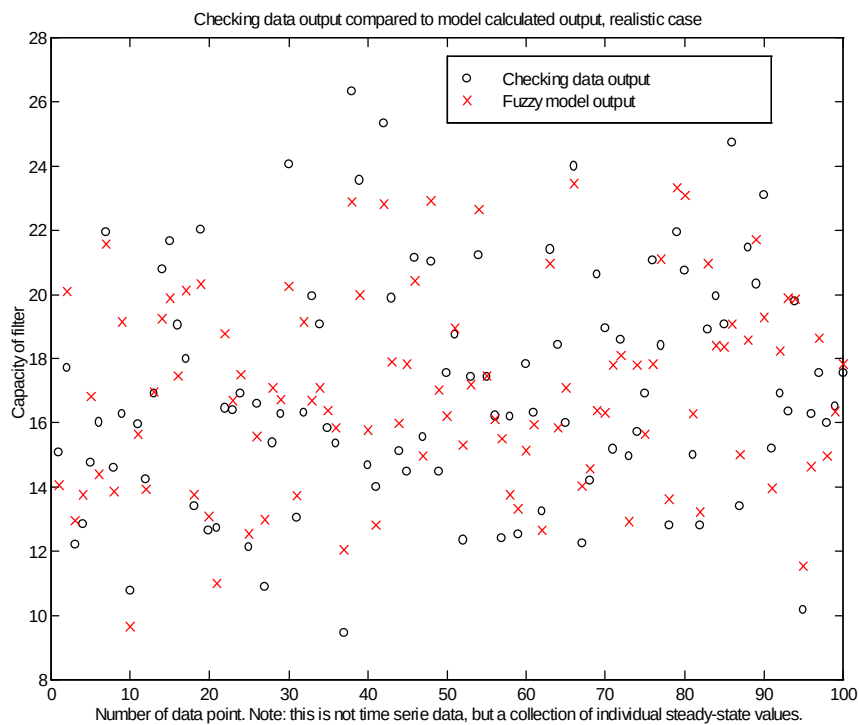


Figure 10. Comparing of checking data and the output of fuzzy model

5. Conclusions

Two different cases of training and checking data sets were generated. The first case was mathematically ideal in the sense, that the output was affected only known and perfectly measured input variables and both the structure and parameters of the rotating disk filter process were constant. Tested fuzzy logic (ANFIS) and neural network (feedforward neural network trained by Levenberg-Marquard optimization) methods were capable to model the ideal nonlinear data almost perfectly without any difficulties.

The second case data was more close to real industrial process data. White noise was added to output of the model and two randomly varied extra non-measured input variables were affecting into output of the model (or if viewed in alternative way, two parameters of the process were time-varying). Fuzzy logic and neural network methods gave both quite similar results: rough forecasting was still possible by models, but the modeling errors were considerably increased.

Both tested methods can be used to model the nonlinear steady-state rotating disk filter data. To achieve better results with real industrial process, detailed process knowledge have to be used to make separate models or expert inference systems for taking care about varying disk and cake resistance, the time-varying parameters which are the difficult to measure on-line.

Acknowledgements

The authors wish to thank Academy of Finland for financial support.

Notation

$$C_{slurry} = \frac{\text{mass of solids in slurry [kg]}}{\text{volume of water in slurry [m}^3\text{]}}$$

L = level of slurry from axis (downwards = negative)

$\dot{m}_{cake.dry}$ = mass flow of dry cake before scrapers

Δp = pressure difference over the cake and disk [Pa or kPa]

r = radius [mm or m]

r_{outer} = outer radius of disk [mm or m]

r_{inner} = inner radius of disk [mm or m]

R_{disk} = resistance of disk [1 / m]

α = spesific resistance of the cake [m / kg]

μ = viscosity of filtrate water [kg / m / s]

ω = angular speed of rotating disk [rad / s]

References

1. Juditsky, A., *Nonlinear Black-box Models in System Identification*, Automatica, Vol. 31, No. 12, pp.1691-1724, 1995
2. Wang, L.-X., *Adaptive Fuzzy Systems and Control*, Prentice Hall, 1994, 232 p.
3. Kanninen, V., *Capacity Model of Disk Filters*, 8th IFAC International Symposium in Mining, Mineral and Metal Processing, Sun City, South-Africa 29-31 August 1995.
4. Demuth, H. and Beale, M., *Neural Network Toolbox for use with Matlab*, The Math Works, Inc., 1994
5. Jang, J.-S. R. and Gulley, N., *Fuzzy Logic Toolbox for use with Matlab*, The Math Works, Inc., 1995.
6. Zurada, J., *Artificial Neural Systems*, West Publishing Company, 1992, 683 p.
7. Boullard, L. , *Application of Artificial Intelligence in Process Control*, Pergamon Press, 1992, 455 p.
8. Reinfrank, M., *An Introduction to Fuzzy control*, Springer Verlag, 1993, 316 p.
9. Jager, R. *Fuzzy Logic in Control*, Doctorate Thesis, Technische Universiteit Delft, 1995, 313 p.
10. Jang, J.-S. R., "ANFIS: Adaptive-Network-based Fuzzy Inference Systems." IEEE Transactions on Systems, Man and Cybernetics, Vol. 23, No. 3, pp. 665-685, May 1993.
11. Jang, J. -S. R. and Sun, C. -T., *Neuro-Fuzzy Modeling and Control*, Proceedings of the IEEE, Vol. 83, No. 3, pp. 378-406, March 1995