DETERMINING THE STRUCTURE OF NONLINEAR MODELS

Jörg Schultz[*], **Stefan Hillenbrand**[†]

[*] *Institut für Regelungs- und Steuerungssysteme, Universität Karlsruhe, Kaiserstr. 12, D-76128 Karlsruhe, Germany*
Fax : +(49) 721/608-5707 e-mail : schultz@irs.etec.uni-karlsruhe.de
[†] *Lehrstuhl für Regelungstechnik und Signaltheorie, Universität Kaiserslautern, Postfach 3049, D-67653 Kaiserslautern, Germany*
Fax : +(49) 631/205-4205 e-mail : hillenbrand@e-technik.uni-kl.de

Keywords : Nonlinear Process Identification, Neural Nets, Modelling, Structure Determination, Input Variable Selection.

Abstract

In order to perform systems analysis or synthesis, it is compulsory to deduce a model of the process. Artificial Neural Networks (ANN) have shown their suitability to identify nonlinear dynamic processes without modelling them theoretically. Since no modelling is performed, the important issue for the Neural Network approach is to determine the required time delays. In this paper, different methods are presented that make it possible to reach this goal. First, some pruning methods are presented to detect non-required input neurons belonging to certain time delays. In order to avoid the high computational efforts of these methods, a new approach is presented which is based on the estimation of the gradient vector of the system nonlinearity. All methods are applied to a continuous-stirred tank reactor.

1 Introduction

The modelling and identification of dynamic processes is necessary in order to perform systems analysis and synthesis. Due to the fact that theoretical modelling of nonlinear dynamic processes might be too difficult or costly, Artificial Neural Networks (ANN) have shown their suitability [1, 2] for the identification of such processes. One approach is to add a delay component to the ANN representing the dynamics (see fig. 1). This structure is called a Time-Delay Neural Network (TDNN).

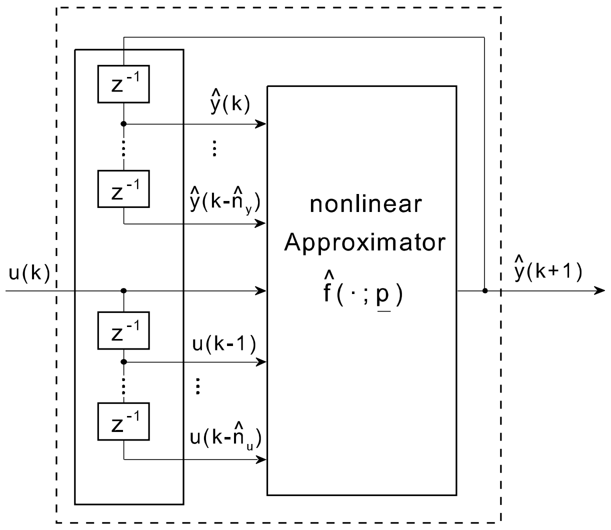

Figure 1: Nonlinear dynamic model that consists of a nonlinear approximator (ANN) and a delay component.

The delay component builds a vector consisting of time delays of the process input and output. This vector is the input of the ANN. Since the model of the process is unknown, the number of delays of the process input and output that is necessary to identify the process has to be determined. This important issue is the subject in this contribution. If too few delays are specified, the TDNN is not able to approximate the process behaviour accurately. Whereas, if the number of delays of the model is too high, the consequence is high computational effort for net training and model validation.

First, a short introduction to the identification of nonlinear dynamic systems using TDNNs is given. After that, pruning methods [3], which are usually used to reduce the number of hidden neurons, are taken into consideration for model structuring in this contribution. Two methods are presented that can be applied to the input neurons.

Moreover, a new method to determine the appropriate set of time delays is presented. It is based on the estimation of the gradient vector of the nonlinearity at different points in the operating domain. In a first step the maximal delay of the output variable is determined. This guarantees a well-conditioned matrix built from process data which is used to estimate the gradient vector. Its elements are taken to find out which of the remaining delays of the input and output can be eliminated. Finally, all methods presented are applied to determine the model structure of a continuous-stirred tank reactor.

2 The identification of a nonlinear dynamic process

A dynamic SISO process

$$y_{k+1} = f\left(y_k, \ldots, y_{k-n_y}, u_k, \ldots, u_{k-n_u}\right) \qquad (1)$$

is to be identified. The process is BIBO-stable in the considered operating domain. The unknown nonlinear mapping $f(\cdot)$ is assumed to be time-invariant, bounded, continuous and differentiable. The maximal time delays that influence the process output y_{k+1} are indicated by n_y (for the output which is fed back) and n_u (for the input variable u_k). The model is represented by the following difference equation

$$\hat{y}_{k+1} = \hat{f}\left(\hat{y}_k, \ldots, \hat{y}_{k-\hat{n}_y}, u_k, \ldots, u_{k-\hat{n}_u}; \underline{p}\right) \qquad (2)$$

with maximal delays $\hat{n}_y$ and $\hat{n}_u$. The vector $\underline{p}$ consists of the model parameters which are, for instance, the weights and biases of a Multi-Layer Perceptron (MLP). They have to be determined in such a way that (2) approximates the input/output behaviour of (1) as well as possible. The usual approach is to minimize the cost function

$$\begin{aligned} J &= \tfrac{1}{2}\textstyle\sum_{k=0}^{N}(y_{k+1}-\hat{y}_{k+1})^2 \\ &= \tfrac{1}{2}\textstyle\sum_{k=0}^{N}\left(y_{k+1}-\hat{f}\left(y_k,\ldots,y_{k-\hat{n}_y},u_k,\ldots,u_{k-\hat{n}_u};\underline{p}\right)\right)^2 \end{aligned} \qquad (3)$$

with regard to the vector $\underline{p}$ using $N+1$ tupels of measured process data. The maximal delays have to be pre-specified as $\hat{n}_u \geq n_u$ and $\hat{n}_y \geq n_y$. All delays between these limits are to be considered since it is unknown which could be discarded. On the other hand, computational efforts in the succeeding steps of the identification process increase with the number of delays since the number of net parameters depends on them. Therefore, the minimal set of delays which is necessary to describe the nonlinear dynamic system sufficiently has to be determined.

3 Pruning methods for structure analysis

Generally, pruning methods [3] are used to locate unnecessary neurons and parameters in an ANN, respectively. Below, two pruning approaches are described to analyse the degree of dependence of the input neurons. All delays up to the assumed maximum $\hat{n}_u > n_u$ and $\hat{n}_y > n_y$ have to be examined.

3.1 Skeletonization

The idea of Skeletonization [4] is to compute how the performance of the ANN changes when a neuron is removed. Using the sum squared error (3) as a measure for the network performance, the relevance of a unit is

$$\rho_i = J_{\text{without unit } i} - J_{\text{with unit } i} \qquad (4)$$

This relevance has to be computed for each input neuron. The smaller ρ_i is, the less impact has the i-th input neuron. The neuron with the smallest ρ_i can be discarded when the corresponding mean approximation error $\frac{1}{N+1}J_{\text{without unit } i}$ due to deletion of neuron i is sufficiently small. The next step is to train the ANN without the deleted neuron and to repeat the steps described above until the mean approximation error of the smallest ρ_i is too large.

3.2 Optimal Brain Damage (OBD)

OBD is a method to find a set of parameters whose deletion will cause the least increase of the objective function (3). In order to gain an insight into the influence of the parameters on the objective function, (3) is expanded by its Taylor series at the determined parameter vector $\underline{p}_{min}$ that minimises (3). A pertubation $\delta\underline{p}$ of the parameter vector $\underline{p}$ will change the objective function (3) by

$$\begin{aligned} \delta J = \textstyle\sum_i g_i \delta p_i + \tfrac{1}{2}\sum_i h_{ii}\delta p_i^2 + \\ + \tfrac{1}{2}\textstyle\sum_{i\neq j} h_{ij}\delta p_i \delta p_j + O\left(\|\delta\underline{p}\|^3\right). \end{aligned} \qquad (5)$$

Here, the δp_is are the components of $\delta \underline{p}$, the g_is are the components of the gradient of J with respect to $\underline{p}$ and the h_{ij}s are the elements of the Hessian matrix of J with respect to $\underline{p}$.

Since it is too difficult to handle (5) analytically, in [5] some simplifications are introduced:

- δJ is well described by a quadratic function at $\underline{p}_{min}$, i.e. $O(\|\delta \underline{p}\|^3) \approx 0$.
- Since the parameter vector $\underline{p}$ is at a minimum, the gradient vector can be neglected ($g_i \approx 0$).
- It is assumed that only the diagonal elements h_{ii} of the Hessian matrix have to be considered in (5).

Thus, J can be approximated [5] as a quadratic function of $\Delta \underline{p} = \underline{p} - \underline{p}_{min}$. The increase of the net error ΔJ depending on the parameter changes Δp_i reduces to

$$\Delta J \approx \frac{1}{2} \sum_{i=1}^{n_p} h_{ii} \Delta p_i^2 \qquad (6)$$

with $n_p = dim(\underline{p})$. The Hessian elements h_{ii} in (6) can be computed efficiently using the Marquardt-Levenberg approximation [6]. If the Marquardt-Levenberg optimization method is also used for the training of the net $\hat{f}(\cdot)$, no extra computation is necessary for OBD.

The original OBD procedure computes the saliencies $s_k = \frac{h_{kk}\Delta p_k^2}{2} = \frac{h_{kk} p_k^2}{2}$ for each considered parameter and deletes some low-saliency parameters. After that, the whole procedure is repeated until no more parameters can be deleted. Since the compulsory delays have to be found, only the fan-out parameters of the input neurons have to be considered. In order to shorten the described procedure, the saliencies of an input neuron are computed by the sum of the saliencies of the fan-out parameters of this neuron. The low-saliency neuron is deleted before training starts again.

The disadvantage of both methods presented is that an ANN with an "oversized" input layer has to be trained. A new approach avoiding this is given in the following section.

4 Model Structuring by Linearization

The idea is to determine the compulsory delays using the gradient vector of $\hat{f}(\cdot;\underline{p})$ with regard to the input vector at a point $\underline{x}_p = [y_k, \ldots, y_{k-\hat{n}_y}, u_k, \ldots, u_{k-\hat{n}_u}]_p$ in the operating domain. A delay term can be discarded if the element of the gradient vector that belongs to that delay term is close to zero within the whole operating domain (see fig. 2).

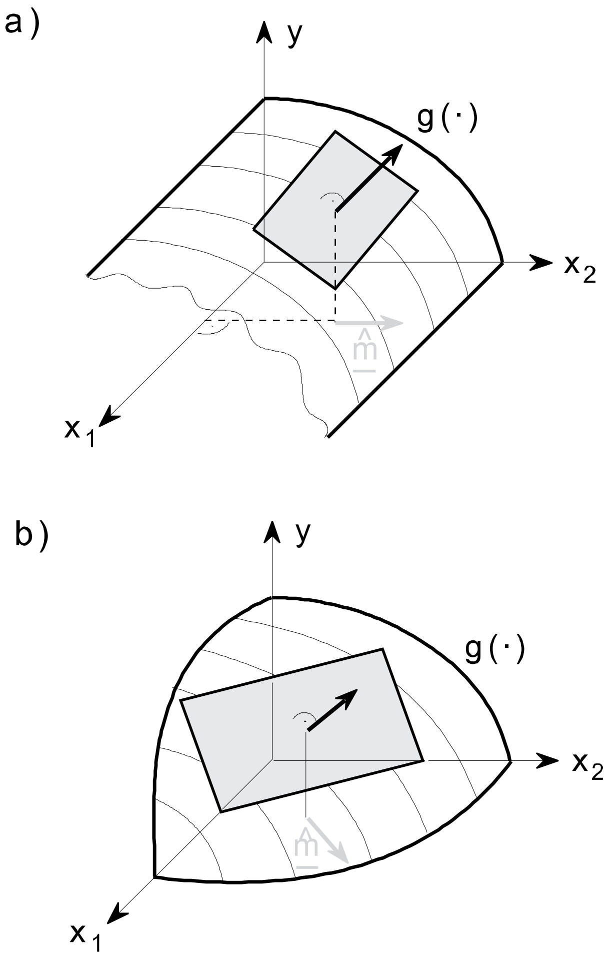

Figure 2: A function $y = g(x_1, x_2)$ is shown in both graphs. In the upper case, the input variable x_1 can be discarded. The element of the vector $\underline{m}$ that belongs to this input variable is zero in the whole input domain. In the lower case, no input variable can be discarded since the vector $\underline{m}$ doesn't hold this condition for each input variable.

In order to avoid the high computational effort to obtain an $\hat{f}(\cdot;\underline{p})$ as in the previous section, measured process data is taken to estimate the gradient vector for a sufficient high number of points $\underline{x}_p$ in the operating domain.

The tangent plane of $\hat{f}(\cdot)$ at $\underline{x}_p$ is given by the Taylor series expansion of $\hat{f}(\cdot)$ at $\underline{x}_p$

$$y_{k+1} = \hat{f}(\underline{x}_p) + \Delta \underline{x}^T \frac{\partial \hat{f}}{\partial \underline{x}}\bigg|_{\underline{x}_p} + O\left(\|\Delta \underline{x}\|^2\right) \quad (7)$$

with $\Delta \underline{x} = \underline{x} - \underline{x}_p$. The vector $\underline{x}$ is built from process data. The remainder can be neglected if $\Delta \underline{x}$ is small enough. This yields

$$\Delta y_{k+1} \approx \Delta \underline{x}^T \frac{\partial \hat{f}}{\partial \underline{x}}\bigg|_{\underline{x}_p} = \Delta \underline{x}^T \underline{\hat{m}}. \quad (8)$$

with the gradient vector $\underline{\hat{m}}$, which has to be estimated by a least-squares method. Using a sufficient number of data N, (8) yields

$$\begin{bmatrix} \Delta y^1_{k+1} \\ \vdots \\ \Delta y^N_{k+1} \end{bmatrix} = \begin{bmatrix} \Delta \underline{x}^T_1 \\ \vdots \\ \Delta \underline{x}^T_N \end{bmatrix} \underline{\hat{m}} + \underline{\hat{e}} = \underline{A}\, \underline{\hat{m}} + \underline{\hat{e}} \quad (9)$$

with

$$\underline{A} = \begin{bmatrix} \Delta y^1_k & \cdots & \Delta y^1_{k-\hat{n}_y} & \Delta u^1_k & \cdots & \Delta u^1_{k-\hat{n}_u} \\ \vdots & & \vdots & \vdots & & \vdots \\ \Delta y^N_k & \cdots & \Delta y^N_{k-\hat{n}_y} & \Delta u^N_k & \cdots & \Delta u^N_{k-\hat{n}_u} \end{bmatrix} \quad (10)$$

and the linearization error $\underline{\hat{e}}$.

In order to obtain a good estimation, it is obligatory that $\underline{A}$ is well-conditioned. This depends not only on the data used but also on the number of considered output delays in (10) as will be shown below.

In the following, it is first outlined how the maximal output delay n_y can be detected. It is assumed that $\Delta n_y = \hat{n}_y - n_y \geq 1$, $\Delta n_u = \hat{n}_u - n_u \geq 1$ and the abbreviation $\Delta_m = min\,(\Delta n_u, \Delta n_y)$ is used. If $f(\cdot)$ were known, its linearization at a point $\underline{x}_p$ would be

$$\begin{aligned} \Delta y_{k+1} = m_1 \Delta y_k + \ldots + m_{n_y+1} \Delta y_{k-n_y} + \\ + m_{n_y+2} \Delta u_k + \ldots + m_{n_y+n_u+2} \Delta u_{k-n_u} + \varepsilon. \end{aligned} \quad (11)$$

Furthermore, the linearizations of (1) by Taylor series expansion after the shifting of the index k by $0 \leq r \leq \Delta_m$ results as:

$$\begin{aligned} \Delta y_{k+1-r} = m^r_1 \Delta y_{k-r} + \ldots + m^r_{n_y+1} \Delta y_{k-n_y-r} + \\ + m^r_{n_y+2} \Delta u_{k-r} + \ldots + m^r_{n_y+n_u+2} \Delta u_{k-n_u-r} + \varepsilon^r \end{aligned} \quad (12)$$

with the linearization error ε^r. The linearizations (12) equally hold with vectors $\Delta \underline{y}^T_{k-\nu} = [\Delta y^1_{k-\nu}, \ldots, \Delta y^N_{k-\nu}]$, $r-1 \leq \nu \leq n_y + r$ and $\Delta \underline{u}^T_{k-\mu} = [\Delta u^1_{k-\mu}, \ldots, \Delta u^N_{k-\mu}]$, $r \leq \mu \leq n_u + r$. All these vectors are built of process data. For reasons of simplification, it is assumed that $\Delta_m = \Delta n_y$. If $\varepsilon^r = 0$, which represents the linear case, it follows from (12) that the vectors $\Delta \underline{y}^T_{k-j} = \left[\Delta y^1_{k-j}, \ldots, \Delta y^N_{k-j}\right]$, $0 \leq j \leq \Delta_m - 1$ are exactly linearly dependent from others $\underline{y}^T_{k-i}$ with $j+1 \leq i \leq j + n_y + 1$ and $\Delta \underline{u}^T_{k-i} = \left[\Delta u^1_{k-i}, \ldots, \Delta u^N_{k-i}\right]$ with $j+1 \leq i \leq j + n_u + 1$. Therefore, as shown in [8], a reduced rank of the matrix $\underline{A}$ in (9) results if $\hat{n}_y > n_y$. Since in the nonlinear case, ε^r cannot be neglected, the column vectors of $\underline{A}$ are not exactly linearly dependent but, in this case, a reduced rank of a matrix $\underline{A}$ can be derived by the ill-conditioning of this matrix. This holds provided that ε^r is sufficiently small, which means (12) is only considered in a small environment of $\underline{x}_p$. $\underline{A}$ is ill-conditioned if $\hat{n}_y > n_y$. This fact is the basis to find the maximal delay of y in the model (2). Therefore, the matrix

$$\underline{A}_\kappa = \left[\Delta \underline{y}_{k-\kappa}, \ldots, \Delta \underline{y}_{k-\hat{n}_y}, \Delta \underline{u}_{k-\kappa}, \ldots, \Delta \underline{u}_{k-\hat{n}_u}\right] \quad (13)$$

is built stepwise for $\kappa = \{\hat{n}_y, \hat{n}_y - 1, \ldots 0, \ldots\}$. If $\hat{n}_y - n_y \leq \kappa \leq \hat{n}_y$, then $\underline{A}_\kappa$ is well-conditioned, since its column vectors are linearly independent. By adding the column vector $\Delta \underline{y}^T_{k+1+n_y-\hat{n}_y}$ for $\kappa = \hat{n}_y - n_y - 1$, it follows with (12) that $\underline{A}_{\hat{n}_y - n_y - 1}$ is ill-conditioned. The index of the last well-conditioned matrix $\underline{A}_\kappa$ is named κ^*. This change from a well- to an ill-conditioned matrix indicates that one more delay is considered than necessary. Thus, when $\kappa^* = \hat{n}_y - n_y$ is detected, the maximal output delay is easily computed by

$$n_y = \hat{n}_y - \kappa^*. \quad (14)$$

It follows that the resulting matrix $\underline{A}_{\hat{n}_y - n_y} = \underline{A}_{\kappa^*}$, that is used to estimate the gradient vector, is well-conditioned only depending on the chosen data. Therefore, it is possible to estimate $\underline{\hat{m}}$. If elements of $\underline{\hat{m}}$ are close to zero, then the corresponding delay terms are negligible at $\underline{x}_p$.

Summing up, the whole procedure works as follows:

1. Select a sufficiently high number of delays in u and y.
2. Compute the conditions $c_\kappa = cond\,\{\underline{A}_\kappa\}$ for $\kappa \leq \hat{n}_y$ step by step until the transition from a well-conditioned matrix to an ill-conditioned matrix

occurs. The maximal output delay is computed by (14) with the detected $\kappa^* = \hat{n}_y - n_y$.

3. Estimate the gradient vector for the determined number of output delays and the included input delays with the last well-conditioned $\underline{A}_{\kappa^*}$.

4. Determine which delay terms are negligible at the considered $\underline{x}_p$ by analysing the corresponding elements of the gradient vector.

This procedure has to be applied to a sufficiently high number of points $\underline{x}_p$. So, the delay terms which are unnecessary in all operating points $\underline{x}_p$ have to be discarded.

The method presented requires that enough data is available in the considered region of $\underline{x}_p$. Moreover, this data has to be distributed in such a way that no linear dependence occurs. Thus, a well-conditioned matrix $\underline{A}_{\hat{n}_y - n_y}$ is guaranteed. These conditions are no limitations. It can always be achieved that the data coming from the input delays is well-distributed since the process input is freely chosen by the operator and the system is assumed to be BIBO-stable and as much data as needed can be generated from the process.

Example: A continuous-stirred tank reactor is to be identified. The inaccessible continuous time model [7]

$$\dot{\underline{x}} = \begin{bmatrix} -0.957 & a_{12}(\underline{x}) \\ -0.323 & a_{22}(\underline{x}) \end{bmatrix} \underline{x} + \begin{bmatrix} 0 \\ 1.548 \end{bmatrix} u$$

with $a_{22}(\underline{x}) = 0.468 \cdot a_{12}(\underline{x}) - 1.815$,

$$a_{12}(\underline{x}) = \begin{cases} \frac{1.05 \cdot 10^{14} \cdot (0.279 - x_1)}{x_2} \left(e^{\frac{-34.289}{1.05 + x_2}} - 6.568 \cdot 10^{-15} \right) & x_2 \neq 0 \\ 21.449 \cdot (0.279 - x_1) & x_2 = 0 \end{cases}$$

and $y(t) = [0 \quad 1]\,\underline{x}(t)$ represents the process. The control variable is the coolant temperature and the output variable is the reactor temperature. The difference equation of the system $y_{k+1} = f(y_k, y_{k-1}, u_k, u_{k-1})$ is obtained by Euler approximation. Thus, $n_y = 1$ and $n_u = 1$.

An ANN with two hidden layers and 6 neurons in each of them is considered. This net is trained using 9818 process patterns until a sufficiently small $J = 5.7 \cdot 10^{-4}$ after 1900 epochs is achieved so that the pruning methods can be applied. The model consists of 91 parameters and its structure is given by $y_{k+1} = \hat{f}(y_k, y_{k-1}, y_{k-2}, u_k, u_{k-1}, u_{k-2}; \underline{p})$. The scaled ρ_is for the model delays are presented in table 1.

ρ_i	y_k	y_{k-1}	y_{k-2}	u_k	u_{k-1}	u_{k-2}
Skel.	0.56	1.0	0.004	1.0	0.67	0.01
OBD	1.0	0.45	0.000	0.42	1.0	0.16

Table 1: The scaled relevance terms that result for the continuous-stirred tank reactor using the skeletonization method and OBD respectively.

The relevance for y_{k-2} and u_{k-2} computed by Skeletonization is quite small. So, one would discard these two terms. In this case, both non-required time delays are detected in one step.

Using OBD one would only discard the delay term y_{k-2}. Apparently, the simplifications made in section 3.2 do not hold in this case.

Thus, a further Neural Network with the input variables y_k, y_{k-1} and u_k, u_{k-1}, u_{k-2} has to be examined. After training to a sufficiently small error with the training data used above, the OBD is applied at this net. The scaled relevances are presented in table 2. This time, OBD yields a sufficiently small relevance of the variable u_{k-2} so that the belonging input term can be neglected. Therefore, the resulting model structure is $y_{k+1} = \hat{f}(y_k, y_{k-1}, u_k, u_{k-1}; \underline{p})$.

ρ_i	y_k	y_{k-1}	y_{k-2}	u_k	u_{k-1}	u_{k-2}
OBD	1.0	0.53	—	1.0	0.41	0.07

Table 2: The scaled relevance terms that result for the continuous-stirred tank reactor using OBD applied at the second Neural Network with 5 input variables.

In conclusion, both methods were successfully applied to select the significant input terms and analysing the model structure. Nevertheless, the computational efforts for the methods are considerable since, first, a net has to be trained so that these methods can be applied.

For the linearization approach the delays y_k, $\ldots, y_{k-2}, u_k, \ldots, u_{k-2}$ are taken into consideration. Thus, $\hat{n}_u = 2$, $\hat{n}_y = 2$. The sequence of matrix conditions of $\underline{A}_\kappa$ is $\{c_2, \ldots, c_{-1}\} = \{1.4, 3.8, 137.5, 1310\}$ at $\underline{x}_{p1}^T = [5.3, 3.3, 2.1, 3.5, 3.4, -1.5] \cdot 10^{-3}$ and $\{c_2, \ldots, c_{-1}\} = \{1.6, 3.5, 1245, 953\}$ at $\underline{x}_{p2}^T =$

[0,0, 0, 0, 0, 0]. The matrix conditions are computed by the MATLAB function *cond* that uses the Singular Value Decomposition. The change from a well- to an ill-conditioned matrix for both $\underline{x}_{p1}$ and $\underline{x}_{p2}$ is detected by the step from $\underline{c}_1$ to $\underline{c}_0$. Thus, $\underline{A}_1$ represents the last well-conditioned matrix and $\kappa^* = 1$. With (14) it follows that $n_y = \hat{n}_y - \kappa^* = 1$. Since no jumping systems are considered using (1), the concluded model structure is $y_k = \hat{f}(y_{k-1}, y_{k-2}, u_{k-1}, u_{k-2}; \underline{p})$. The well-conditioned matrix $\underline{A}_1$ is used to estimate $\underline{\hat{m}}$ at $\underline{x}_{p1}$ and $\underline{x}_{p2}$. These vectors are $\underline{m}_1^T = [2.2, -1.6, 0.16, -0.23]$ and $\underline{m}_2^T = [2.6, -1.7, 0.2, -0.21]$. Thus, y_{k-1}, u_{k-1} and u_{k-2} have to be included in the model. Only one index shift is necessary to get the same structure of model and system. For this approach only data selection, computation of the conditions c_κ, and the estimation of $\underline{\hat{m}}$ has to be carried out. That is a reasonable effort in comparison to the training of a net.

5 Conclusion

For the identification of nonlinear dynamic systems using neural networks with external delays (TDNN), it is necessary to determine the required time delays. First, two pruning methods are presented with which the needed time delays can be found. However, this approach is characterised by a tremendous computational burden. This results from the training of 'oversized' ANNs. For this reason, a new method based on the estimation of the gradient vector at a sufficient number of points in the operating domain is presented. In a first step, the maximal output delay is determined using the condition of a matrix built from process data. Then, the gradient vector is estimated. If an element is close to zero at all considered points, its belonging delay term can be discarded. All methods are applied to the continuous-stirred tank reactor. The presented new approach is suitable to detect the necessary delays with tolerable computing efforts.

References

[1] Hunt, K., Sbarboro, D.: "Neural Networks for Control Systems – A Survey", *Automatica*, 28(**6**), pp. 1083-1112,(1992).

[2] Narendra, K. S., Parthasarathy, K.: "Identification and Control of Dynamical Systems Using Neural Networks", *IEEE Trans. on Neural Networks*, 1(**1**), pp. 4-27, March 90.

[3] Reed, R.:"Pruning Algorithms – A survey", *IEEE Trans. on Neural Networks*, **4**, No. 5, Sept. 1993.

[4] Mozer, M., Smolensky, P.: "Skeletonization: A Technique for trimming the fat from a network via relevance assessment.", *Advances in Neural Information Processing Systems*, **1**, 1988, pp. 107-115.

[5] Cun, Y., Denker, J.: "Optimal Brain Damage", *Advances in Neural Information Processing Systems*, **2**, 1989, pp. 598-605.

[6] Press, W. H.: "Numerical Recipes in C", Cambridge University Press, Second Edition, 1992.

[7] Föllinger, O.: "Nichtlineare Regelungssysteme I,II", R. Oldenbourg, Munich, 7. Edition, 1993.

[8] Lee, R.C.K.: "Optimal Estimation, Identification and Control", MIT Press, 1964.