A&QT-R 2002 (THETA 13)
2002 IEEE-TTTC International Conference on Automation, Quality and Testing, Robotics
May 23 – 25, 2002, Cluj-Napoca, Romania

Model Order Estimation for Noisy Identification

Tibor Laczó, László Sragner

Budapest University of Technology and Economics
Department of Measurement and Information Systems
Magyar tudósok körútja 2., Budapest, HUNGARY
tibur@sch.bme.hu, sragner@sch.bme.hu

Abstract. One of the principal targets of nonlinear system identification is determining the structure of an unknown system, where structure if defined if a model class and the size of the model is selected. In this paper we examine the Lipschitz method to estimate the size of the model. During modelling we cannot neglect the problems of noise effects. This affects the goodness of model negatively and the result of the Lipschitz method may be incorrect. To avoid or reduce the effect of noisy data we use the Errors-In-Variables (EIV) cost function what reduces the effects of the noise by correcting the input data. The paper proposes a new method to combine EIV cost function and the Lipschitz algorithm to get a better estimate of the model order. The usefulness of this idea is justified by extensive experiments.

Keywords: Lipschitz-method, Errors-In-Variables, Noisy data, Order estimation, Dynamic system

1 Introduction

The modelling of an unknown system is a difficult problem because we have information only from the behaviour of the system in the past. If the system cannot be defined with exact mathematical formulas, we need a black-box model and we can predict the future behaviour from the input-output data in the past. If we work with dynamic systems the order of the data is important, because in this case the earlier states of the system affects the later outputs.

To identify a dynamic system we have to choose the class of the model and determine the order of the model. This means that how many past inputs and outputs are used in the calculation of the current output.

There are several criteria e.g. the Akaike Information Criteria (AIC) [1] and the Minimum Description Length (MDL) [2], what can be used to qualify a model. Usually they built up of two terms: the first one is calculated from the output error of the model and the second one penalises the model complexity. Model validation based on information criteria needs the setting of the models parameters what is the most time consuming step in the system identification. We will use the Lipschitz method, which estimates the order of the model directly from the input-output data without the need of building the

model [5]. Lipschitz method, however, is rather sensitive to measurement noise, which means that we cannot get definite estimate for model order if the is based on noisy data

The problem of the output noise can be managed if we can assume that the noise is a Gaussian process with zero mean value. This case can be managed by avoiding overfitting. The other problem is the input noise. It can also be presumed that the noise is a Gaussian process with zero mean value. During the nonlinear transformation of the input to the output we transmit the noise through the model and we get a noise in the output, which is not additive even if the input noise was additive. To manage the effects of input noise we use the Errors-In-Variables cost function what corrects the input data during the estimation of the model's parameters [4].

The main task of this paper is to examine the results of using the two algorithms together in noisy environment. The main idea is that using EIV method we can reduce the noise at the input data; these improved data will be used for the Lipschitz method to get a better estimate of model order. In Section 2 we review the tools used in the modelling problem. In Section 3 we describe shortly the Lipschitz algorithm, what can estimate the optimal complexity of the model from the input-output sequence without creating the model. In Section 4 the Errors-In-Variables (EIV) cost function is presented, what uses statistical information to correct inputs during the training of the model. In Section 5 we show how this two methods can be used together, and some experimental results are presented on an artificial and on a real life industrial problem. (LD-converter modelling in steelmaking) [6].

2 Background

For the model construction we use Multi Layer Perceptrons (MLPs) neural networks. It can approximate any continuous nonlinear functions and with a little modification it can be used for dynamic modelling. The easiest way to form dynamic network from a static one is the to use a tapped delay lines at the input and the output signal paths. This is a so-called NOE model where the static network's inputs are connected to the actual input, the earlier inputs and the earlier outputs. The network's parameters are trained by backpropagation (BP) algorithm using Least-Squares (LS) or EIV cost function. The greatest difference between dynamic and static network validation and training is that in the static case we can qualify the network for each sample point, in the dynamic case, however, we have to qualify it through a whole input sequence. The behaviour of the network can be qualified with the mean of the squared error of the output of a time-interval and the modification of the parameters are made according to this.

3 The Lipschitz quotient

While we create the dynamic model from static mapping, we have to define how many earlier inputs and outputs are used. This is a pair of numbers what is named as the order of the model. The next method is able to give an estimate for the order of a model without calculating its parameters [5].

The main idea of the Lipschitz quotient is that a continuous process's output does not vary independently between two known points. If a system has the continuity property we can estimate the maximum of the gradients only from the input-output data. This

maximum is an important characteristic of the transfer function of the system. MLP also has the continuity property so we can reconstruct the input-output data of a NARX [7] model from the systems measured input-output data. The transformation of the model is a simple $R^{m*b+(l+1)*a} \rightarrow R^b$ mapping where (m,l) is the presumed order of the examined model, b is the dimension of the output space and a is the dimension of the input space. We calculate a quotient from the gradient between training point *t1* and *t2* is:

$$L_{t1,t2}^{(m,l)} = \sqrt{mb+(l+1)a} \sqrt{\frac{\left\| y_{t2} - y_{t1} \right\|^2}{\frac{1}{mb}\sum_{p=1}^{m} \left\| y_{(t2)-p} - y_{(t1)-p} \right\|^2 + \frac{1}{(l+1)*a}\sum_{q=0}^{l} \left\| u_{(t2)-q} - u_{(t1)-q} \right\|^2}} \qquad \textbf{(1)}$$

The maximum of this quotient is $L(m,l)$. If (m_0,l_0) is the optimal order than $L(m_0\text{-}1,l_0)$ and $L(m_0,l_0\text{-}1)$ is much larger than $L(m_0,l_0)$ and $L(m_0+1,l_0)$ and $L(m_0,l_0+1)$ are near to $L(m_0,l_0)$. Our job is to find the sharpest breakpoint in the graph of $L(m,l)$ *vs.* (m,l) [5]. If the points are noisy we suppose that the maximum is spread around a typical value so we use geometrical average. This reduces the noise but the graph will not have clear breakpoint. This makes the estimation of the order difficult and its precision is strongly depends on the amount of noise.

4 EIV (Errors-In-Variables) cost function

Although the EIV cost function originally was used in linear system identification, it is also usable for nonlinear transfer function modelling [3]. Using the EIV approach we correct the input data according to the variance of the noise in parallel with the training of the network. The theory applies for MIMO (Multiple Inputs Multiple Outputs) systems, however, to simplify the notations we discuss it for SISO (Single Inputs Single Outputs) systems. We seek a black-box model for the system $y_k^{[i]}(\Theta) = f_{NN}(u_k^{[i]}, \Theta)$ where Θ is the parameter vector of the neural network, $u_k^{[i]}$ are the measured inputs, $y_k^{[i]}(\Theta)$ are the networks outputs for these inputs [4]. The training of the network is done with the set of $N \times M$ input-output samples in which each measurement is repeated M-times. The input and output matrices are denoted by **U** and **Y**. For the k-th measurement the sample variances of the inputs and outputs ($\hat{\sigma}_{u,k}^2$ and $\hat{\sigma}_{y,k}^2$ respectively), the sample covariance matrix $\hat{\sigma}_{uy,k}^2$ and the mean values $\bar{u}_k$ and $\bar{y}_k$ can be calculated or can be estimated [4].

If $M=1$, so we have only one measurement in every sample points, we have to estimate the variances of the noise. The knowledge of the variances on the inputs and outputs allows the use of the EIV cost function [4].

$$C_{EIV} = \frac{M}{N}\sum_{k=1}^{N}\left(\frac{(\bar{y}_k - f_{NN}(u_k,\Theta))^2}{\sigma_{y,k}^2} + \frac{(\bar{u}_k - u_k)^2}{\sigma_{u,k}^2} \right) \qquad \textbf{(2)}$$

u_k is the true but unknown input value, which must be estimated. The formula weights the error of the input and the output according to the reciprocal of the noise's variance. This means that the more noisy data points are will have a smaller role in the modelling than the less noisy points. It can be assumed that if we train the network with

noiseless data and with the EIV cost function the global optimum can be found. It can be proved that with noisy data it also converges to the true model's parameters.

During the EIV training we modify not only the weights, but the input data as well, so this leads to an increased risk of being trapped in a local minimum during the optimisation because of the danger of overfitting (the number of free parameters of the optimisation is increased). To avoid overfitting as a starting step we train the network with LS cost function before the EIV approach is applied.

In the case of large variances, the measurements are typically grouped together related to the horizontal axis, allowing for an easy, but inaccurate mapping of the data. The overfitting, and inaccurate mapping can successfully be prevented by the use of an early-stopping algorithm. This needs a validation set of samples.

5 The joint use of the two algorithms, some experimental results

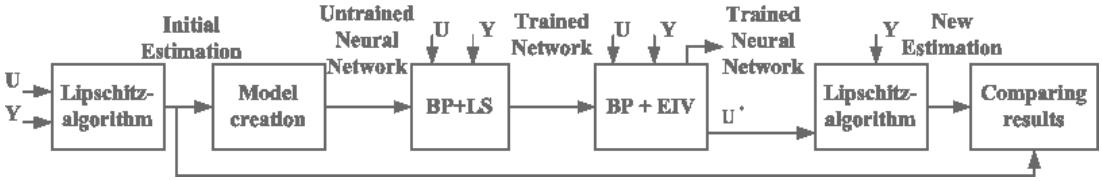

Fig. 1. The steps of the experiments

First we estimate the order of the dynamic model with the Lipschitz method. Because of the noise we probably do not get sharp breakpoint in the results. In that case we have to choose an order what is likely to be good enough. In the next step we generate a neural network with the estimated order. This network first trained with the BP algorithm and the LS cost function. When the training reaches an error limit we stop the training with LS and use the EIV cost function. We control the overfitting with a test sequence. If the neural network is trained we use the Lipschitz algorithm again and compare its result with the previous values. The transfer function of the examined system is: (where y_j^i means the *j*-th co-ordinate of y_{t-i})

$$y_1^0 = \frac{6y_1^1y_2^1u_1^1u_2^1u_3^1y_1^2y_2^2u_2^2u_3^2y_1^3y_2^3y_2^3 + 5\left(y_1^1y_2^2u_1^2u_2^2y_1^3 + 3.5\right)}{1+\left(y^1\right)^T y^1 + \left(u^1\right)^T u^2 + \left(y^2\right)^T u_{1,2}^2 + \left(y_1^2\right)^2 + \left(y^2\right)^T y^3 + \left(y_1^2\right)^3 + \left(u^2\right)^T u^2}$$

$$y_2^0 = \frac{8y_1^1y_2^1u_1^1u_3^1y_1^2u_1^2u_2^2y_1^3y_2^3y_2^3 - 6\left(y_2^1u_3^1y_2^2u_2^2u_3^2y_1^3y_2^3 + 3.5\right)}{1+\left(y^1\right)^T y^1 + \left(u^1\right)^T u^2 + \left(y^2\right)^T u_{2,3}^2 + \left(y_1^2\right)^2 + \left(y^2\right)^T y^3 - 2\left(y_2^2\right)^3 + \left(u^2\right)^T u^2} \qquad (3)$$

First we generate training and test sequences, which describe the system's transfer function satisfactory. Second we run the algorithm on the original noiseless training sequence to examine whether the Lipschitz algorithm gives the optimal (3,2) order for the given system. We could see that the decrease in the direction of the feedforward is more radical than in the other direction. This happens because the tested system depends stronger from its input values than the feedback ones, and also the feedback values are smaller than the input ones.

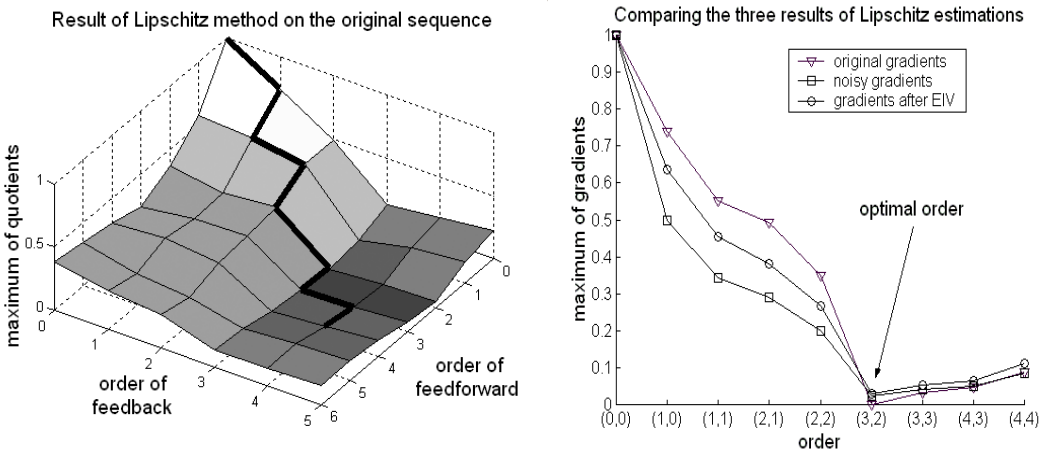

Fig. 2. The first figure is the maximum of the quotients for all orders. The second figure is the result of three different runs of the Lipschitz method on the selected orders. The selected orders are marked with the solid line on the first figure.

We can see that the algorithm gives good estimates. The test sequence has been normalised to make it commensurable with the next examinations. In the next step we add noise to the system. This is an additive Gaussian noise with variance 0.005. On the second figure we can see the effect of the noise. The estimation on the true order of (3,2) becomes less sharp because the gradients what the algorithms calculate vary around its noiseless values and we have to use an averaging (geometrical average). Despite of that the true order is also shown on the graph. The next step is to create the network and estimate the parameters using standard BP. We train 250 epochs and we stop learning at 0.05 error. After that we start to use the EIV cost function and while we have the original points we check the measure of correction during the training. If we see that the error on the input values stop decreasing overfitting is occur and we stop training. This could be eliminated also with dividing the input sequence to independent testing and training sequence, but now to check the algorithm properly we use the original points. The EIV lowers the error from 0.05 to 0.04 in 31 epochs and the error on the inputs is reduced from 14.48 to 14.43. Here we stop the training because the error on the inputs starts to increase. We can see that the correction on the inputs is very little only 0.05. Despite that on the third figure we can see that the graph of the Lipschitz quotients changes radically. This is because the algorithm searches for the greatest gradients and at this point a little change causes great changes. The estimation of the Lipschitz algorithm keeps the correct estimation. For conclusion we can say that it is useful to use the two algorithms together, they can help each other to work properly. The experiments, however, justify this result only partly. For the artificial problem the effect of the joint application is significant, however for the industrial only rather slightly improvement can be reached, which is probably caused by the inaccurate estimate of the variances.

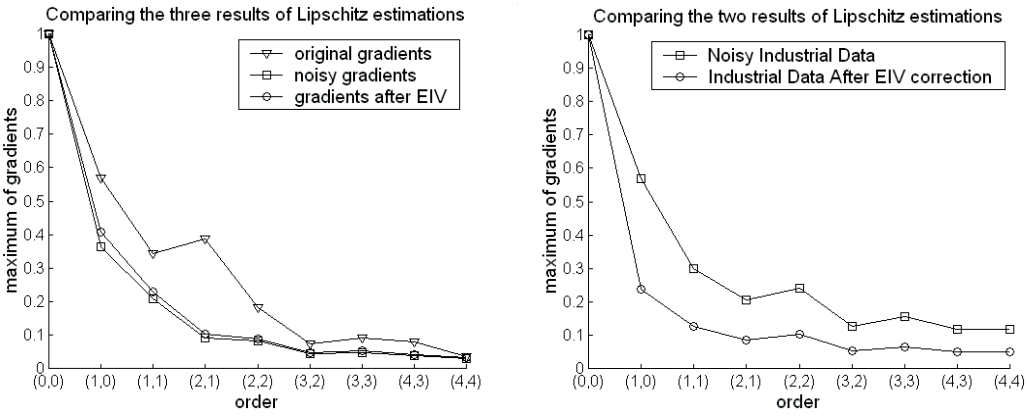

Fig. 3. On the first figure we add more noise to the original points. On the second figure we can see the results of the estimations on the industrial data.

We next add more noise (variance=0.008) to the original points of the same artificial problem. We can see that the noisy curve has no sharp breakpoint. After the EIV there is still no breakpoint but some sharpening has happened. The next phase is to try to use the methods on a real world industrial process (modeling of a steelmaking LD-converter) [6]. In this problem we have only noisy data from the process. The first estimates using Lipschitz method show that the system's order is (2,1) or (3,2). Because the data are rather noisy and because there is no prior information of the noise variances we have to estimate them. These estimates, however, are probably not accurate enough because of the lack of enough information. So EIV method cannot improve significantly the performance of the Lipschitz method. Further works are needed to estimate the noise's parameters and to modify the method to be less sensitive to measurement noise.

6 References

1. H. Akaike, "A new look at the statistical model identification", IEEE Transaction on Aut. Control Vol. AC-19, No.6., 1974, pp.716-723
2. J. Rissanen, "Modeling by Shortest Data Description", Automatica, Vol. 14., pp.465-471
3. G. Vandersteen, "Identification of Linear and Nonlinear Systems in an Errors-In-Variables Least Squares and Total Least Squares Framework", PhD thesis, Vrije Universiteit Brussel, Belgium, April 1997
4. J. Van Gorp, J. Schoukens, R. Pintelon, "The Errors-In-Variables Cost Function for Learning Neural Networks with Noisy Inputs", ANNIE 1998, Intelligent Engineering Systems Through Artificial Neural Networks, Volume 8,pp. 141-146.
5. X. He, H Asada, "A new method for identifying orders of input-output models for nonlinear dynamic systems" Proceedings of the American Control Conference, San Francisco, California, June 1993, pp. 2520-2523.
6. P. Berényi, G. Horváth, B. Pataki, Gy. Strausz, "Hybrid-Neural Modelling of a Complex Industrial Process", IEEE Instrumentation and Measurement Technology Conference, Budapest, 2001.
7. K. S. Narendra, K. Pathasarathy, "Identification and Control of Dynamical Systems Using Neural Networks", IEEE Trans. Neural Networks, Vol.1.pp. 4-27,1990.