Estimation of grey box and black box models for non-linear circuit datas

Lennart Ljung, Qinghua Zhang, Peter Lindskog, Anatoli Juditski

Division of Automatic Control

E-mail: ljung@isy.liu.se, zhang@irisa.fr,
peter.lindskog@niradynamics.se,
Anatoli.Iouditski@inrialpes.fr

14th June 2007

Report no.: [LiTH-isy-R-2792](#)

Accepted for publication in Proc. NOLCOS 2004 - IFAC Symposium on Nonlinear Control Systems, Stuttgart

Address:

Department of Electrical Engineering
Linköpings universitet
SE-581 83 Linköping, Sweden

WWW: <http://www.control.isy.liu.se>

AUTOMATIC CONTROL
REGLERTEKNIK
LINKÖPINGS UNIVERSITET

Abstract

Identification of non-linear systems is a challenge, due to the richness of both model structures and estimation approaches. As a case study, in this paper we test a number of methods on a data set collected from an electrical circuit at the Free University of Brussels. These methods are based on black box and grey box model structures or on a mixture of them, which are all implemented in a forthcoming Matlab toolbox. The results of this case study illustrate the importance of the use of custom (user defined) regressors in a black box model. Based on physical knowledge or on insights gained through experience, such custom regressors allow to build efficient models with a relatively simple model structure.

Keywords: identification

ESTIMATION OF GREY BOX AND BLACK BOX MODELS FOR NON-LINEAR CIRCUIT DATA

Lennart Ljung* Qinghua Zhang** Peter Lindskog***
Anatoli Juditski****

* *Div. of Automatic Control, Linköping University, SE-58183
Linköping, Sweden, e-mail: ljung@isy.liu.se*

** *IRISA-INRIA, Rennes, France zhang@irisa.fr*

*** *NIRA Dynamics, Linköping, Sweden, pl@niradynamics.se*

**** *LMC, Univ. Grenoble, France*

Anatoli.Iouditski@inrialpes.fr

Abstract Identification of non-linear systems is a challenge, due to the richness of both model structures and estimation approaches. As a case study, in this paper we test a number of methods on a data set collected from an electrical circuit at the Free University of Brussels. These methods are based on black box and grey box model structures or on a mixture of them, which are all implemented in a forthcoming MATLAB toolbox. The results of this case study illustrate the importance of the use of custom (user defined) regressors in a black box model. Based on physical knowledge or on insights gained through experience, such custom regressors allow to build efficient models with a relatively simple model structure.

Keywords: non-linear system identification, black box model, grey box model, Matlab toolbox.

1. INTRODUCTION

Identification of non-linear systems is a challenge, due to the richness of both model structures and estimation approaches. It is therefore valuable to gain more experience about how different methods perform in practice. In this contribution we shall test a number of methods on a data set from an electrical circuit (“the silver box”), collected at the Free University of Brussels.

The methods we will investigate are all implemented in a forthcoming MATLAB toolbox, “The non-linear system identification toolbox”, which is an add-on to The MathWork’s System Identification toolbox, (Ljung, 2003). We will occasionally indicate some commands from that toolbox to pinpoint the actual model estimations. The toolbox covers both traditional and less traditional non-linear black-box models for function

and dynamic model estimation, such as neural networks, wavenets, and trees. See (Sjöberg *et al.*, 1995) for the general black-box framework. The toolbox also covers estimation of models of Wiener and/or Hammerstein type, i.e. linear dynamic blocks in conjunction with non-linear static mappings. Moreover, structured user defined grey box non-linear models with parameters to be estimated are also included. As a mixture of grey and black box models (“dark-grey models”) the toolbox allows the definition of “Custom regressors” to be included both in linear and non-linear structures. This feature proves to be especially valuable in the current context.

2. MODELS AND METHODS

Predictor Models

We shall denote the measured system output at time t by $y(t)$ and the input by $u(t)$. For simplicity we assume a unit sampling interval. The data record from 1 to t will be denoted by

$$Z^t = \{y(1), u(1), y(2), u(2), \dots, y(t), u(t)\} \quad (1)$$

For the black and dark-grey box world, the basic model is in *predictor* or *innovations* form

$$y(t) = f(\theta, \varphi(t)) + e(t) \quad (2)$$

$$\varphi(t) = h(Z^{t-1}, u(t)) \quad (3)$$

Note that $\varphi(t)$ cannot depend on $y(t)$, but may depend on $u(t)$ (like in the “silver box” case study of this paper). Typically models used for controller design have at least one pure input delay, then it is simply written $\varphi(t) = h(Z^{t-1})$. In case

$$h(Z^{t-1}, u(t)) = [y(t - n_p), \dots, y(t - n_p - n_a + 1), u(t - n_k), \dots, u(t - n_k - n_b + 1)] \quad (4)$$

we call (2) an $([n_a, n_b, n_k, n_p])$ *NLARX model*.

However, arbitrary user defined functions of past data are also allowed:

$$h(Z^{t-1}, u(t)) = [h_1(Z^{t-1}, u(t)) \dots h_n(Z^{t-1}, u(t))]$$

Then (2) will be called an *NLARX model with Custom Regressors* $h_k(Z^{t-1}, u(t))$.

The predictor function

All the above holds for arbitrary functions f , which are functions from $\mathcal{R}^n$ to $\mathcal{R}^p$, where n is the dimension of $\varphi(t)$ and p is the number of outputs. Such functions have been extensively discussed in the literature, see, e.g. (Sjöberg *et al.*, 1995; Juditsky *et al.*, 1995). A common case is that f is parameterized as

$$f(\theta, \varphi) = \sum_{k=1}^d \alpha_k \kappa(\beta_k(\varphi - \gamma_k)) \quad (5)$$

$$\theta = \{\alpha_k, \beta_k, \gamma_k, k = 1, \dots, d\} \quad (6)$$

Depending on the choice of κ and the interpretation of its argument, this covers among many choices *one hidden layer neural networks*, *wavenet models*, *radial basis networks*, *neuro-fuzzy models*, *piece-wise linear models*, *local kernel models*, etc. See (Sjöberg *et al.*, 1995). Other common parameterizations of f include *binary trees*, where θ corresponds to the branching rules in the nodes.

Prediction and Simulation

The innovations model (2) will directly give rise to a one-step predictor formula, by simply omitting

the term $e(t)$. So with known past input-output data Z^{t-1} it is immediate to compute the *one-step ahead prediction* for a given parameter value θ :

$$\hat{y}(t|\theta) = f(\theta, \varphi(t)) \quad (7)$$

It is worth while to stress how the model (2) is *simulated*, i.e. how to compute noise-free outputs $y_s(t)$ from an input sequence $\{u(1), \dots, u(t)\}$:

$$y_s(t, \theta) = f(\theta, \varphi_s(t, \theta)) \quad (8a)$$

$$\varphi_s(t, \theta) = h(Z_s(\theta)^{t-1}, u(t)) \quad (8b)$$

$$Z_s(\theta)^{t-1} = \{y_s(1, \theta), u(1), y_s(2, \theta), u(2), \dots, y_s(t-1, \theta), u(t-1)\} \quad (8c)$$

This holds for all kinds of regressors. The calculation of *k-step ahead predictors* for (2) is somewhat more complex, and discussed, e.g. in (Zhang and Ljung, 2004).

If we use (8a) also as a way to compute the predicted output $\hat{y}(t|\theta)$, that is all past measured outputs are ignored in the predictor expression, we call the corresponding model a *nonlinear output error* (NLOE) model.

Model Estimation

The basic method for estimating θ is to minimize the prediction error, e.g. (Ljung, 1999). Depending on whether we use an NLARX or NLOE model, this gives two different expressions:

$$\hat{\theta}_N = \arg \min_{\theta} \sum_{t=1}^N (y(t) - \hat{y}(t|\theta))^2 \quad (9a)$$

$$\hat{\theta}_N = \arg \min_{\theta} \sum_{t=1}^N (y(t) - y_s(t, \theta))^2 \quad (9b)$$

See (Zhang, 2004) for a discussion how to deal with the minimization of (9b) in the nonlinear case.

3. THE DATA

The models and methods will be applied to a set of real data collected from an electric circuit, described in Section 3.4.6 of (Pintelon and Schoukens, 2001) and (Schoukens *et al.*, 2003). The circuit theoretically obeys

$$m \frac{d^2 y(t)}{dt^2} + d \frac{dy(t)}{dt} + ay(t) + by(t)^3 = u(t) \quad (10)$$

A data record of 131072 samples was collected by J. Schoukens and R. Pintelon. It is shown in Figure 1. The data set was detrended and the first 40495 samples were used for validation and samples from 40586 to 127410 for estimation. The sampling interval is 0.0016384 sec.

Figure 1. The validation set, corresponding to “the arrow head” (40495 samples), followed by the estimation data set (86825 samples)

4. RESULTS

The basic regressor orders were chosen as $n_a = 2$, $n_b = 3$, $n_k = 0$ and $n_p = 1$ in (4). First the linear ARX and model models (corresponding to linear f) were computed using (9a) and (9b):

```
m1 = arx(ze,[2 3 0]);
mlo = oe(ze,[3 2 0]);
```

(comments on the commands of the MATLAB toolbox: ze is the data object containing the estimation data set, $[2 \ 3 \ 0]$ means $n_a = 2$, $n_b = 3$, $n_k = 0$, whereas $n_p = 1$ by default. For output error models, the numbering is different: $[3 \ 2 \ 0]$ for an output error model means the same number of “poles” and “zeros” as $[2 \ 3 \ 0]$ for ARX-models).

Then non-linear black-box models with a tree nonlinearity, a one hidden layer sigmoidal network and a wavenet nonlinearity, respectively, were estimated:

```
mt = nlarx(ze,[2 3 0], 'tree');
ms = nlarx(ze,[2 3 0], 'sigmoid');
mw = nlarx(ze,[2 3 0], 'wavenet');
```

For the tree and wavenet nonlinearities, the size is by default chosen automatically with a generalized cross validation (GCV) criterion. The default size of the one hidden layer sigmoidal network is 10 neurons. Different number of neurons in the network can be tested by

```
ms50 = nlarx(ze,[2 3 0], 'sigmoid', ...
    'number',50);
ms75 = nlarx(ze,[2 3 0], 'sigmoid', ...
    'number',75);
```

These were computed using (9a).

For the sigmoidal network, iterative minimization of this criterion is used, consisting of Gauss-Newton and/or Levenberg-Marquardt steps. For trees and wavelets a “one-shot” procedure is used

to attempt to minimize this function. The wavelet network is can be seen as a radial-basis network with a particular way to seed the dilation and location parameters. Further iterations to minimize (9a) could be obtained by

```
mwf = nlarx(ze,mw, 'maxiter',10)
```

but such results are not shown in this article.

4.1 Output Error Variants

An output error variant of the sigmoid and wavenet model is obtained by

```
mso = nloe(ze,[3 2 0], 'sigmoid');
mwo = nloe(ze,[3 2 0], 'wavenet');
```

corresponding to the minimization (9b). Different number of neurons for the sigmoidal network can be tested in an obvious way:

```
ms50o = nloe(ze,[3 2 0], 'sigmoid', ...
    'number',50);
ms75o = nloe(ze,[3 2 0], 'sigmoid', ...
    'number',75);
```

Figure 2. A scatter plot of $e(t)$ against $y(t-1)$.

The accuracy of all these models evaluated on the validation data set will be summarized in the table at the end of this section.

4.2 Custom Regressors

Now a question is whether the accuracy can be improved by introducing custom regressors. To find good candidates for custom regressors, the residuals from the linear model were matched against past y and u , like in (ze and e are data objects defined in MATLAB for which $ze.y$ and $e.y$ allow to access their output data field)

```
e = pe(ze,m1);
plot(ze.y(1:end-1),e.y(2:end),'.')
```

This plot, which was the most clear one of these kinds is shown in Figure 2. It clearly indicates a third order relationship

$$e(t) \sim y(t-1)^3$$

For that reason a linear-in-parameters model with this custom regressor added was tried out (`y1` is the default output name used by the toolbox):

```
mlc = nlarx(ze,[2 3 0], 'linear', ...
    'custom',{'y1(t-1)^3'});
```

That gave the model

$$\begin{aligned} y(t) = & 1.4724y(t-1) - 0.93357y(t-2) \\ & + 0.048916(t) + 0.37088u(t-1) \\ & + 0.045163u(t-2) - 1.4712y(t-1)^3 \end{aligned}$$

As seen from the table, the inclusion of this custom regressor improved the RMS value by almost a factor of 10, showing that this effect is the single most important nonlinearity. A natural question is whether the model be improved by using these regressors (including the custom one) in a sigmoidal neural network. We try with both 10 and 75 nodes in a one hidden layer network:

```
msc = nlarx(ze,[2 3 0], 'sigm', ...
    'custom',{'y1(t-1)^3'});
msc75 = nlarx(ze,[2 3 0], 'sigm', ...
    'custom',{'y1(t-1)^3'}, 'number', 25);
msc75 = nlarx(ze,msc75); % More iterations
```

Similarly a tree model is estimated;

```
mtc = nlarx(ze,[2 3 0], 'tree', ...
    'custom',{'y1(t-1)^3'});
```

For tree models various structures where certain regressors enter linearly can be tested efficiently:

```
mta = nlarx(ze,[2 3 0], 'tree', ...
    'LinearReg','auto');
```

It picks out as the “best” structure the one where $y(t-1)$ enters nonlinearly in f and all the others linearly (in other words, f is the sum of a non-linear function of $y(t-1)$ and a linear regression of the other regressors). This result is well in line with the findings of a custom regressor.

4.3 More regressors

While order tests indicate a small improvement by increasing the model orders beyond `[2 3 0]`, only very marginally improved simulation fit is obtained by increasing the model orders using linear models, both with and without the custom regressor $y^3(t-1)$. This shows that regressors of the type $y(t-k)$, $u(t-k)$, $k > 3$ have very little to offer when entering linearly in the model. It could of course still happen that the could be of value as a non-linear contribution. For that reason we investigate higher order models of the kind

```
ms8c50 = nlarx(ze,[8 8 0], 'sigm', 'custom', ...
    {'y1(t-1)^3'}, 'number', 50)
```

This actually gave significant improvements, as seen in Table 1.

4.4 Semi-physical and physical models

Since combinations of linear dynamic blocks and nonlinear static ones turned out to be successful in (Schoukens *et al.*, 2003) we tried a Hammerstein-Wiener model with static nonlinearities of neural network sigmoid types on the input as well as the output sides of a linear system:

```
mhw = nlhw(ze,[3 2 0], 'unonlin', 'sigm', ...
    'OutputNonlin','sigm');
```

Finally a file `silver.c` that implements a continuous time state space description of (10) was written (in C-code) to constitute a non-linear grey-box model and its physical parameters were estimated:

```
mg = nlgrey('silver', Nom_par)
mg = pem(ze,mg)
```

To evaluate the models we compute the simulation errors for the validation data (cf (8a))

$$e = zv.y - \text{sim}(m, zv.u)$$

Note that a pure simulation is obtained: the model output depends on past inputs only. The errors are plotted in figure 3 for the linear model `m1` and the nonlinear model with custom regressor `ms10c30` respectively. The RMS value of e (i.e $\sqrt{\frac{1}{N} \sum_{t=1}^N e^2(k)}$) for the 18 different models are given in Table 1.

5. CONCLUSIONS

Estimating nonlinear black box models involves several choices: (1) The regressors, (2) which regressors enter linearly and which enter nonlinearly in f (3) The predictor function f , (4) The size/flexibility of f . One may easily end up with an overwhelming amount of different models.

One may note a few differences regarding the choice of f : Estimating trees is typically the fastest method, but the model may be large to store. Wavenet models can automatically choose the number of units in the model (by cross validation), while sigmoidal network models often give better results when the number of neurons has been well tuned, and the initialization has been successful. In theory, all these nonlinearities are universal approximators, so which one is best will in general depend on the data.

The figures and the table in the previous section summarize our findings in the present case. It is

Figure 3. The simulation error for the linear model **ml**(left) and the nonlinear model **ms10c30** (right). The maximum absolute error is 0.2539 and 0.0064, resp.

Model	RMS	Model	RMS
ml	0.0144	mlo	0.0152
mt	0.0074	mta	0.0036
mw	0.0070	mwo	0.0062
ms	0.0042	mso	0.0026
ms75	0.00052	ms75o	0.00046
mlc	0.0017	mtc	0.0083
mwc	0.0020	msc	0.0013
msc75	0.00038		
ml20c	0.0016	ms5c50	0.00039
ms8c50	0.00037	ms10c30	0.00030
mwh	0.0139	mg	0.0133

Table 1. Root mean square values of the simulation error for the different investigated models. The first group, above the first divider all use regressors $y(t-1), y(t-2), u(t), u(t-1), u(t-2)$. A trailing 'o' indicates that the model has been computed by the output error technique (9b). The second group also uses the custom regressor $y^3(t-1)$. The third

group uses regressors $y(t-1), \dots, y(t-r), u(t), \dots, u(t-r), y^3(t-1)$, where r is the number following the first two letters in the model (i.e. 20, 5, 8, 10). The second letter (**l**, **t**, **w**, **s**) denotes the type of nonlinearity (linear, tree, wavelet or sigmoidal neural network). The trailing number (75, 50, 30) denotes the number of neurons in the corresponding one hidden layer sigmoidal neural network.) The last group of models contains the physically and semi-physically parameterized models.

interesting to note that it is of great importance to include the custom regressor $y(t-1)^3$. Even in a linear-in-parameter model (**mlc**) this gives a great improvement, that is not easy to obtain by general black-box non-linearities. It requires high flexibility of f ("many neurons") to capture the cubic non-linearity. One of the best results

(**msc75**) is achieved by using this custom regressor together with a general non-linear function. By using output error models with many neurons, similar accuracy can also be obtained, but one should note that estimating **ms75o** takes at least 5 times longer than estimating **msc75**.

It is also of interest to note by comparing **mw/ms** and **mwo/mso** that there is a clear advantage to use the output error fitting method (9b) for obtaining good models for simulation. It would had been interesting to include the custom regressor in **mso**, but the current software for (9b) does not support custom regressors.

It is interesting to note that due to the large amount of estimation data, estimation of quite complicated models can be supported. The RMS errors for the validation data are dominated by the last 4000 values, which correspond to larger input amplitudes than present in estimation data. It is thus a matter of extrapolation of the data in the model. Quite interestingly, the complex models with many neurons and many regressors significantly reduce the maximum errors in this part of the validation data. This shows that there is "hidden" information in the estimation data about how the system performs also at higher input amplitudes.

One might also note that the result for the Hammerstein-Wiener model **mwh** is not so impressive. That shows that the double linear system in the "feedback loop" of (Schoukens *et al.*, 2003) is essential.

Finally, the non-linear grey box model did not so well. A reason could be that the actual non-linear device used in the experiment deviates a bit from the theoretical model structure (10), or that the model estimation was caught in a local minimum.

The methods employed in this study all work with an arbitrary amount of data. With the many estimation data, the estimation time ranges from less than a second for linear-in-parameters model to a few hours for the most complex models on a laptop. All computational code was implemented in MATLAB m-files.

6. REFERENCES

- Juditsky, A., H. Hjalmarsson, A. Benveniste, B. Delyon, L. Ljung, J. Sjöberg and Q. Zhang (1995). Nonlinear black-box modeling in system identification: Mathematical foundations. *Automatica* **31**(12), 1724–1750.
- Ljung, L. (1999). *System Identification - Theory for the User*. 2nd ed.. Prentice-Hall. Upper Saddle River, N.J.
- Ljung, L. (2003). *System Identification Toolbox for use with MATLAB. Version 6..* 6th ed.. The MathWorks, Inc. Natick, MA.
- Pintelon, R. and J. Schoukens (2001). *System Identification – A Frequency Domain Approach*. IEEE Press. New York.
- Schoukens, J., J. G. Nemeth, P. Crama, Y. Rolain and R. Pintelon (2003). Fast approximate identification of nonlinear systems. *Automatica* **39**(7), 1267–1274. July.
- Sjöberg, J., Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.Y. Glorennec, H. Hjalmarsson and A. Juditsky (1995). Nonlinear black-box modeling in system identification: A unified overview. *Automatica* **31**(12), 1691–1724.
- Zhang, Q. (2004). Nonlinear system identification with output error model through stabilized simulation. In: *Proc. NOLCOS 2004*. Stuttgart, Germany. Submitted.
- Zhang, Q. and L. Ljung (2004). Multiple steps prediction with nonlinear arx models. In: *Proc. NOLCOS 2004*. Stuttgart, Germany. Submitted.

Avdelning, Institution Division, Department Division of Automatic Control Department of Electrical Engineering		Datum Date 2007-06-14
Språk Language ☐ Svenska/Swedish ☒ Engelska/English ☐ _____	Rapporttyp Report category ☐ Licentiatavhandling ☐ Examensarbete ☐ C-uppsats ☐ D-uppsats ☒ Övrig rapport ☐ _____	ISBN _____ ISRN _____ Serietitel och serienummer ISSN Title of series, numbering 1400-3902
URL för elektronisk version http://www.control.isy.liu.se		LiTH-ISY-R-2792
Titel Estimation of grey box and black box models for non-linear circuit datas Title		
Författare Lennart Ljung, Qinghua Zhang, Peter Lindskog, Anatoli Juditski Author		
Sammanfattning Abstract Identification of non-linear systems is a challenge, due to the richness of both model structures and estimation approaches. As a case study, in this paper we test a number of methods on a data set collected from an electrical circuit at the Free University of Brussels. These methods are based on black box and grey box model structures or on a mixture of them, which are all implemented in a forthcoming Matlab toolbox. The results of this case study illustrate the importance of the use of custom (user defined) regressors in a black box model. Based on physical knowledge or on insights gained through experience, such custom regressors allow to build efficient models with a relatively simple model structure.		
Nyckelord Keywords identification		