

Identification of Nonlinear Systems

Lennart Ljung

Division of Automatic Control

E-mail: ljung@isy.liu.se

14th June 2007

Report no.: [LiTH-ISY-R-2784](#)

Accepted for publication in Proc. 9th International Conference on Control, Automation, Robotics and Vision, ICARCV 2006

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 nonlinear systems is a problem with many facets and roots in several diverse fields. It is not possible to survey the area in a short text. The current presentation gives a subjective view on some essential features in the area. These concern a classification of methods, the use of physical insight in models, and some overall issues like bias-variance trade-off. It is also discusses how the methods can be made available in software packages.

Keywords: identification, non-linear systems

Identification of Nonlinear Systems

Lennart Ljung Div. of Automatic Control
Linköping University
Linköping, Sweden
ljung@isy.liu.se

Abstract— Identification of nonlinear systems is a problem with many facets and roots in several diverse fields. It is not possible to survey the area in a short text. The current presentation gives a subjective view on some essential features in the area. These concern a classification of methods, the use of physical insight in models, and some overall issues like bias-variance trade-off. It also discusses how the methods can be made available in software packages.

I. INTRODUCTION

To construct and estimate models on non-linear dynamic systems is an important and difficult task. It relies upon many different disciplines: Physical modeling, e.g. (Fritzson, 2004), (Bohlin, 2006), mathematical statistics, e.g. (Hastie *et al.*, 2001), neural network techniques, e.g. (Barron, 1989), learning theory and support vector machines, e.g. (Vapnik, 1998), (Suykens *et al.*, 2002), automatic control and system identification, e.g. (Sjöberg *et al.*, 1995), (Ljung, 2006), and several others. Nonlinear models play important roles in many different application fields, and many specific problem areas have developed their own techniques and nomenclatures. As a consequence, there is a pronounced proliferation of methods, concepts and results, and it is not so easy to orient oneself in the area.

It is not possible to give a short, comprehensive survey of the field, and many highly relevant papers and results will not be discussed here. Rather, the paper gives my own subjective views on a few issues that I consider to be central for the estimation of nonlinear models.

II. THE BASIC STATISTICAL PROBLEM

Most basic ideas from system identification, choice of model structures and model sizes are brought out by considering the basic curve fitting problem from elementary statistics: There is an unknown function $g_0(x)$. For a sequence of x -values (regressors) $\{x_1, x_2, \dots, x_N\}$ (that may or may not be chosen by the user) observe the corresponding function values with some noise:

$$y(k) = g_0(x_k) + e(k) \quad (1)$$

The problem is to construct an estimate

$$\hat{g}_N(x) \quad (2)$$

from

$$Z^N = \{y(1), x_1, y(2), x_2, \dots, y(N), x_N\} \quad (3)$$

This is a well known basic problem, that many people have encountered already in high-school. In most applications, x is a vector of dimension, say, d . This means that g defines a

surface in R^{d+1} if y is scalar. If $y(k)$ itself is a p -dimensional vector, it is in this perspective convenient to view the problem as p separate surface-fitting problems, one for each component of y . The construction of the function estimate $\hat{g}_N(x)$ involves several issues that will be treated in this paper, primarily for the application to dynamic system models:

- What are the regressors x in system identification? (Section III-A)
- How to classify the various approaches to find the estimate? (Section III-B)
- What are the user questions involved in the choice of method? (Section III-C)
- Basic issues in non-parametric and parametric methods (Section IV and Sections V – VII)
- How to make methods and algorithms available and accessible in software? (Section VIII)

III. SOME OVERALL ISSUES

A. Choice of Regressors for Dynamic Systems

When models of dynamic systems are sought k would be a time index and the raw observation data are sequences of inputs $u(t)$ and outputs $y(t)$:

$$Z_{yu}^N = \{y(1), u(1), y(2), u(2), \dots, y(N), u(N)\} \quad (4)$$

The task is then to provide a model that is capable of predicting future outputs from past observations:

$$\hat{y}(t|t-1) = g(Z_{yu}^{t-1}, t) \quad (5)$$

To cast this formulation into the curve-fitting framework (1)-(3) we need first to define what are the regressors x . This is a problem related to the choice of states in a state space representation of the system. In general, the regressors are chosen as finite-dimensional projections of past data:

$$x_t = \varphi(t) = \varphi(Z_{yu}^{t-1}) \quad (6)$$

We shall return to a discussion of regressor choices in Section V-D, but for the time being, think of regressors as being a finite collection of past inputs and outputs:

$$x_t = \varphi(t) = [y(t-1), \dots, y(t-n_a), u(t-1), \dots, u(t-n_b)]^T \quad (7)$$

See, among many references, e.g. (Billings, 1990), (Nelles, 2001) for accounts that specifically deal with dynamic systems. Generally speaking, many issues that relate to methods and algorithms do not really depend on the nature of x . This means that the niche for dynamic systems applications is less pronounced in the nonlinear case than in the linear systems case.

B. Characterization of Methods and Models

The large family of methods for estimation of nonlinear models can be classified along several different aspects. In many cases the dividing line is not sharp, but it is anyway a useful way to get a grip on the various possibilities. Below we list a number of contrasting terms that can be used to characterize models and methods for nonlinear identification. The different terms are in no way orthogonal – indeed they are typically quite correlated.

Parametric vs. Nonparametric Methods: A *parametric method* is one that forms a family of candidate function descriptions

$$\mathcal{G} = \{g(x, \theta) | \theta \in D \subseteq R^n\} \quad (8)$$

parameterized by an n -dimensional parameter vector θ . The search for the function (2) is then carried out in terms of θ , typically by optimizing some criterion. The resulting parameter estimate $\hat{\theta}_N$ then gives the estimated function by

$$\hat{g}_N(x) = g(x, \hat{\theta}_N) \quad (9)$$

A *nonparametric method* does not explicitly work with parameterized function families. It typically forms $\hat{g}_N(x)$ by averaging over relevant measurements of y_k . In some cases the distinction may be difficult.

Global vs. Local Methods: A *global method* basically uses all data in (3) to form the estimate $\hat{g}_N(x)$ for any x . A *local method* only uses observation pairs $\{y_k, x_k\}$ with x_k in a local neighborhood of x . Nonparametric methods are typically local.

Regression vs. Classification Problems: In a *regression problem* the task is to estimate a function g as described above. In a *classification problem* the task is to classify observations x into two or more distinct classes. The latter problem can be described as a function estimation problem, by letting the range space of g be discrete, assuming as many values as there are classes. Therefore there are many similarities between classification and regression problems, but the area of classification or *pattern recognition* has several unique features, see, e.g. (Fukunaga, 1990).

Black-box vs. Grey-box Models: A *black-box model* is estimated from data without using any specific insights into how the data were generated. A *grey-box model* is estimated using some ideas about the character of the process that generated the data. Now, there are many different variants, depending on the level of insights used. See Sections VI–VII.

Off-the-Shelf Models vs. Models-on-Demand: An *off-the-shelf model* is estimated from all data and put on the shelf to deliver its value for any x , whenever asked for. A *model-on-demand* is computed from the data (3) at a particular value x only when this function value has come in demand.

Batch Methods vs. Recursive Methods: A *batch method* uses all data Z^N to compute the estimate $\hat{g}_N(x)$. A *recursive method* condenses the information in the data Z^N into a fixed-dimensional vector R_N and computes the estimate $\hat{g}_{N+1}(x)$ from $\hat{g}_N(x)$, R_N and $\{y(N+1), x_{N+1}\}$. For recursive methods, k typically is a time index.

C. The Basic Choices

Bias-variance Trade-off: The objective if of course to find a model $\hat{g}_N(x)$ that is as close as possible to $g_0(x)$. If the disturbances $e(k)$ in (1) are thought of as random variables, $\hat{g}_N(x)$ is a random variable, and a natural measure of size or the error is the mean square error (MSE)

$$\begin{aligned} M_N(x) &= E(g_0(x) - \hat{g}_N(x))^2 \\ &= B_N(x)^2 + W_N(x) \end{aligned} \quad (10a)$$

$$B_N(x) = g_0(x) - g_N^*(x), \quad (10b)$$

$$g_N^*(x) = E\hat{g}_N(x) \quad (10c)$$

$$W_N(x) = E(g_N^*(x) - \hat{g}_N(x))^2 \quad (10d)$$

where the MSE is split into the *bias error* $B_N(x)$ and the *variance error* $W_N(x)$. The symbol E denotes mathematical expectation w.r.t. $e(\cdot)$. A typical case is that g_N^* does not depend on N and that we have

$$\hat{g}_N(x) \rightarrow g^*(x) \text{ as } N \rightarrow \infty \quad (11)$$

In order to make the MSE small, we like both the bias error and the variance error to be small. All methods for estimating \hat{g}_N have in one way or another some knob to tune the estimate. This knob turned one way will decrease bias as variance is increased, and vice versa, when turned the other way. A crucial problem is to find the best trade-off in this tuning. Note that the trade-off as such may depend on the function argument x . One may pick a particular x for the tuning or look at some average over x .

One such average is over the regressors that were used in the collected data:

$$\bar{M}_N = \frac{1}{N} \sum_{k=1}^N M_N(x_k) \quad (12a)$$

$$\bar{W}_N = \frac{1}{N} \sum_{k=1}^N W_N(x_k) \quad (12b)$$

Choice of Norms: A typical parametric method uses a least squares criterion of fit to select the parameter estimate:

$$\hat{\theta}_N = \arg \min_{\theta} V_N(\theta) \quad (13a)$$

$$V_N(\theta) = \frac{1}{N} \sum_{k=1}^N \ell(y(k) - g(x_k, \theta)) \quad (13b)$$

$$\ell(\varepsilon) = \varepsilon^2 \quad (13c)$$

The choice of norm $\ell(\varepsilon)$ can be any measure of size, not necessarily a quadratic norm.

Regularization: When the dimension of θ is large, it turns out to be useful to add a term to the criterion (13a):

$$\hat{\theta}_N = \arg \min_{\theta} V_N(\theta) + r(\theta) \quad (14a)$$

where $r(\theta)$ is a *regularization* term that somehow penalizes large/bad values of θ . A typical form is

$$r(\theta) = \delta \|\theta - \theta^\# \|^2 \quad (14b)$$

Here $\theta^\#$ is a value toward which the parameters are adjusted, typically 0. The *regularization parameter* δ is then a knob that will control the bias-variance trade-off.

Sparseness of Data: The nonlinear estimation problem with a d -dimensional x can be seen as a surface-fitting problem in R^{d+1} . Now, even for moderately large d , this is a huge space. There are several ways to illustrate this. Consider for example the unit cube in R^{d+1} , i.e. $\{x; |x_k| \leq 1 \forall k\}$. Even with a moderate resolution of 0.2 along each coordinate, it takes 10^{d+1} small cubes with side length 0.2 to fill the unit cube. To describe a surface in the unit cube the required amount of data to have at least one observation in each small cube is overwhelming even for $d = 5$. The observed data set (3) will by necessity be very sparse in the space where the surface is going to be estimated.

Local Optima: Typical optimization techniques to find the estimate employ algorithms like (13). The minimization can seldom be done by closed form expressions (essentially only when $\ell(\varepsilon) = \varepsilon^2$ and $g(x, \theta)$ is linear in θ .) The search for the minimum is then typically carried out by iterative local search, like Gauss-Newton algorithms. They can only guarantee convergence to a *local optimum*, while it is the global one that is the target. This could be a serious problem in several methods.

Validation and Generalization: There is a saying that you can draw an elephant if only given four parameters (and make it wag its tail with one more.) The meaning is that it is not so impressive that you can reproduce observed behavior by adjusting a model to the observations. The real test comes when you have to use your model to reproduce new, fresh data. This is the essence of *model validation* (nowadays often called *model generalization*). The traditional statistical term is *cross validation*. See among many references, e.g. (Golub *et al.*, 1979).

To be able to describe the outcome of an experiment before it has been carried out is clearly a very good and convincing quality aspect of the model used. Such cross validation techniques are often at the heart of methods that determine the bias-variance trade-off.

IV. NONPARAMETRIC FUNCTION APPROXIMATION

According to (1), the function values are observed in additive noise. If many observations were made for the same value of x_k it would thus be possible to estimate $g_0(x_k)$ by averaging over the corresponding $y(k)$. This is the basic idea behind nonparametric methods: To average over relevant observations $y(k)$ to form an estimate of the function at a particular value x . A general reference to nonparametric regression is (Härdle, 1990).

A. Kernel Methods

The averaging or smoothing of observations takes the basic form

$$\hat{g}_N(x) = \sum_{k=1}^N w_k y(k) \quad (15a)$$

$$\sum_{k=1}^N w_k = 1 \quad (15b)$$

The weights w_k will depend both on the target point x and the observation point x_k :

$$w_k = C(x, x_k) \quad (16a)$$

Typically, they depend only on the distance between the two points:

$$C(x, x_k) = \frac{K_h(x - x_k)}{\sum_{j=1}^N K_h(x - x_j)} \quad (16b)$$

$$K_h(\tilde{x}) = K(\tilde{x}/h) \quad (16c)$$

where h is a parameter that scales the function K . This is an example of a *kernel method*, more precisely the *Nadaraya-Watson estimator*, (Nadaraya, 1964). Typical choices of the kernel function K are

$$K(\tilde{x}) = \frac{1}{\sqrt{2\pi}} e^{-\tilde{x}^2/2} \text{ (Gaussian)} \quad (17a)$$

$$K(\tilde{x}) = \frac{3}{4} \max\{1 - \tilde{x}^2, 0\} \text{ (Epanechnikov)} \quad (17b)$$

If the kernel is (essentially) zero for $|\tilde{x}| > 1$, observations that are further away than h (the *bandwidth*) from the target point x in (15) will not be used in the function estimate.

It is obvious that the bandwidth parameter in this case is what controls the bias-variance trade-off: A small bandwidth gives few data to average over and hence a large variance. A large bandwidth means that the averaging takes place over a large area, where the true function may change quite a bit, thus leading to large bias.

B. Local Polynomial Methods

In a kernel estimator, the function value is estimated as a mean over a local neighborhood. A more sophisticated approach would be to compute a more advanced estimate within the neighborhood. For example, the function could be approximated as a polynomial within the chosen neighborhood. The coefficients of the polynomial are computed using a weighted least squares fit, the weights typically chosen as a kernel $K_h(u)$, (16c)-(17), giving more weight to the observations close to the target value x . The estimate $\hat{g}_N(x)$ would then be this polynomial's value at x . This is the *local polynomial method*, see, e.g. (Fan and Gijbels, 1996). Clearly, the Nadaraya-Watson estimator corresponds to a local polynomial approach with polynomials of zero order. It also follows that the local polynomial method is closely related to *local composite models*, (Section VII-C), often used in control applications.

C. Direct Weight Optimization

A very direct approach to determine the weights in a nonparametric estimator (15) would be to choose them so that the MSE $M_N(x)$, (10a), at the target point x , is minimized w.r.t. w_k . To carry out the minimization, the true function $g_0(x)$ needs to be known. To handle that, first a maximization of the MSE is carried out w.r.t. a function family \mathcal{G} that g_0 is

assumed to belong to:

$$\hat{g}_N = \sum_{k=1}^N w_k y(k) \quad (18a)$$

$$\sum_{k=1}^N w_k = 1 \quad (18b)$$

$$w_k = \arg \min_{w_k} \max_{g_0 \in \mathcal{G}} M_N(x) g \quad (18c)$$

This method is described in (Roll *et al.*, 2005). The result depends, of course, on the function family \mathcal{G} . For example, if \mathcal{G} is chosen to be a parametric family of functions, like (8), linearly parameterized in θ , the resulting estimate is (naturally enough) the least squares estimate (9). If, on the other hand, the family consists of Lipschitz continuous functions

$$\mathcal{G}_2(L) = \{g(x); |g(x_1) - g(x_2)| \leq L|x_1 - x_2|\} \quad (19)$$

the resulting estimate (18) is a kernel type estimator, typically with the Epanechnikov kernel, and a bandwidth that is automatically selected from L , the assumed noise level, and the available observations. See also (Sacks and Ylvisaker, 1978).

V. BLACK-BOX PARAMETRIC MODELS

A. Basis Function Expansion

In a black-box setting the idea is to parameterize the function $g(x, \theta)$ in a flexible way, so that it can well approximate any feasible true functions $g_0(x)$. A typical choice is to use function expansion

$$g(x, \theta) = \sum_{k=1}^m \alpha_k g_k(x) \quad (20a)$$

with some basis functions g_k .

Scalar Regressor Case.: It turns out that a powerful choice of basis functions is to let them be generated from one and the same “mother function” $\kappa(x)$ and scale and translate it according to

$$g_k(x) = \kappa(\beta_k(x - \gamma_k)) \quad (20b)$$

For example, with $\kappa(x) = \cos(x)$ this gives a Fourier transform expansion with β and γ corresponding to frequency and phase. A more typical example is given by the unit pulse $\kappa(x) = U(x)$

$$U(x) = \begin{cases} 1 & \text{if } 0 \leq x \leq 1 \\ 0 & \text{else} \end{cases} \quad (21)$$

The parameter γ will place this unit pulse anywhere along the real axis, and β will give an arbitrary width to it. The expansion (20) will then describe any piecewise constant function. This, in turn, can approximate any reasonable functions arbitrarily well for large enough m . Clearly a similar result is obtained if κ is chosen as the kernels in (17). This illustrates the approximation power of the choice (20). If κ is chosen as a step, or a soft step

$$\kappa(x) = \sigma(x) = \frac{1}{1 + e^{-x}} \quad (22)$$

the conclusions are similar.

Several Regressors.: It is convenient to let κ be a function of a scalar argument, even in case x is a vector, and interpret the argument $\beta(x - \gamma)$ accordingly. Three interpretations are commonly used:

Radial: Interpret $\beta(x - \gamma)$ as $\|x - \gamma\|_\beta$ with $\|x - \gamma\|_\beta^2 = (x - \gamma)^T \beta (x - \gamma)$, (β being a psd matrix) so that the argument is constant over ellipsoids.

Ridge: Interpret $\beta(x - \gamma)$ as $\beta^T x - \gamma$ with β a column vector and γ a scalar. Then the argument is constant over hyperplanes.

Tensor: Interpret κ is a product of factors corresponding to the components of the vector: $\kappa(\beta(x - \gamma)) = \prod_{k=1}^d \kappa(\beta_k(x_k - \gamma_k))$. γ and β are d -dimensional vectors and subscript denotes component.

B. Examples of Named Structures

There is a very extensive literature on black-box linear models of the kind just described. Many terms and names and derivations from different starting points have been used. Among the most commonly used terms we have (cf (Sjöberg *et al.*, 1995), (Ljung, 1999), ch 5):

- **ANN: Artificial Neural Networks**

- The common one hidden layer Sigmoidal Neural Networks use the sigmoid basic function (22) and the ridge extension to higher regressor dimensions.
- The **Radial Basis Networks** use radial regressor extension, typically with the Gaussian basic function (17a).

- **Least Squares Support Vector Machines**, (Suykens *et al.*, 2002), are derived using an argument in an abstract feature space, but in action they have many features in common with radial basis neural networks with fixed scale and location parameters.

- The **wavelet** expansion of a function is obtained with κ as the “mother wavelet” and double indexing (over j and k) in the sum (20) with $\beta_j = 2^j$ and $\gamma_k = 2^{-j}k$ as fixed choices. The wavenet structure, (Zhang and Benveniste, 1992) is based on an initial wavelet expansion, suppressing of small $\alpha_{k,j}$, followed by a possible refinement of scale and location parameters.

- So called **(Neuro)-Fuzzy modeling**, (Jang and Sun, 1995), (Harris *et al.*, 2002), is based on fuzzy modeling: Signal levels are characterized by fuzzy logic, and numerical values are adjusted to data. This corresponds to (20) with κ being the membership functions and with tensor expansion to higher regressor dimensions.

Linear Regressions: With fixed scale and location parameters β and γ , the expansion (20) will be a linear regression. This makes the estimation of α a linear least squares problem, and is an often used special case, e.g., (Suykens *et al.*, 2002), and (Harris *et al.*, 2002).

C. Simulation and Prediction

A model of a dynamical system can be used both for simulation and prediction. It is important to realize the distinction

between these uses, and we shall here define it for the simplest case.

Suppose the regressor is $x_t = \varphi(t) = [y(t-1), u(t-1)]^T$. The (one-step ahead) predicted output at time t for a given model θ is then

$$\hat{y}_p(t|\theta) = g([y(t-1), u(t-1)]^T, \theta) \quad (23)$$

It uses the previous measurement $y(t-1)$.

A tougher test is to check how the model would behave in simulation, i.e., when only the input sequence u is used. The simulated output is obtained as above, by replacing the measured output by the simulated output from the previous step:

$$\hat{y}_s(t, \theta) = g([\hat{y}_s(t-1, \theta), u(t-1)]^T, \theta) \quad (24)$$

Notice that this simulation algorithm is a dynamical system. It could very well show instability, even if the predictor (23) is stable. It is in general difficult to analyze the stability properties of (24).

D. Choice of Regressors

We can now return to a more detailed discussion on how to choose regressors for a dynamical model. There are essentially four players:

- Outputs $y(t-k)$, Inputs $u(t-k)$
- Simulated model outputs $\hat{y}_s(t-k, \theta)$
- Predicted model outputs $\hat{y}_p(t-k|\theta)$

as defined above.

Regressors for dynamical systems are often chosen among those. In analogy with linear models (e.g., (Ljung, 1999), Section 4.2) they can be named as follows (see also (Billings, 1990)):

- NLFIR-models use past inputs
- NLARX-models use past inputs and outputs
- NLOE-models use past inputs and past simulated outputs
- NLARMAX-models use inputs, outputs and predicted outputs
- NLBJ-models use all four regressor types

VI. PARAMETRIC MODELS WITH SUBSTANTIAL PHYSICAL INSIGHT

Grey-box models incorporate in some way physical insights. Models with lightest shade of grey are obtained by diligent and extensive physical modeling, resulting in a model of fixed structure, but with physical parameters of unknown or uncertain numerical values.

A. Physical Modeling: DAEs

Modern object oriented modeling tools, like MODELICA, ((Fritzson, 2004)) do not necessarily deliver the resulting model in state space form, but as a collection of differential algebraic equations (DAE):

$$F_k(\xi(t), \dot{\xi}(t), z(t), w(t), \theta), k = 1, \dots, K \quad (25)$$

Here z are measured signals, being inputs and outputs, but not necessarily distinguished as such. w are unmeasured

disturbance signals, possibly modeled as stochastic processes. θ are the unknown physical parameters. ξ are internal variables that are used to describe the dynamic relationships.

The nonlinear identification problem is to estimate θ from the measured $z(t)$. In general, this is a difficult problem, that has not yet been treated in full generality. A good reference for a deterministic setting is (Schittkowski, 2002). Identification of nonlinear DAEs in a stochastic setting is discussed in (Gerding, 2006).

Identifiability of models that are given as sets of (polynomial) DAEs is treated in (Ljung and Glad, 1994).

B. State-space Models

If the model equations can be transformed into a state space form

$$\dot{x}(t) = f(x(t), u(t), \theta) \quad (26a)$$

$$y(t) = h(x(t), u(t), \theta) + w(t) \quad (26b)$$

where w is white noise, a formal treatment is possible: For each parameter θ this defines a simulated (predicted) output $\hat{y}(t|\theta)$ which is the parameterized function

$$\hat{y}(t|\theta) = g(Z_{yu}^{t-1}, \theta)$$

in somewhat implicit form. Minimizing a criterion like (13) will then actually be the Maximum Likelihood method. This really requires w to be white measurement noise. Some more sophisticated noise modeling is possible, usually involving *ad hoc* nonlinear observers.

The approach is conceptually simple, but could be very demanding in practice, since the minimization problem will take substantial effort and the criterion may have several local minima.

A recent approach using the EM-method, for the case where f and h in (26) are affine in θ is described in (Schön *et al.*, 2006). Particle filter techniques to deal with Maximum Likelihood methods to identify nonlinear systems are described in (Andrieu *et al.*, 2004).

VII. PARAMETRIC MODELS WITH SOME PHYSICAL INSIGHT

Models with darker shades of grey typically result after a more leisurely modeling work.

A. Semi-physical Modeling

By *semi-physical modeling* we mean to find nonlinear transformations of the measured data, so that the transformed data stand a better chance to describe the system in a linear relationship. The basic rule for this process (to ensure its leisurely aspect) is that only high-school physics should be required and the work must take no more than 10 minutes.

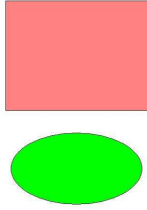
To give a trivial example, consider a process where water is heated by an immersion heater. The input is the voltage applied to the heater, and the output is the temperature of the water. Any attempt to build a linear model from voltage to temperature will fail. A moment's reflection (obeying the rules of semi-physical modeling) tells us that it is the power

of the heater that is the driving stimulus for the temperature; thus let the squared voltage be the input to a linear model generating water temperature at the output. Despite the trivial nature of this example, it is good to keep as a template for data preprocessing. Many identification attempts have failed, due to lack of adequate semi-physical modeling. See, e.g., (Ljung, 1999), Examples 5.1 and pages 533 - 536 for more examples of this kind.

B. Block-oriented Models

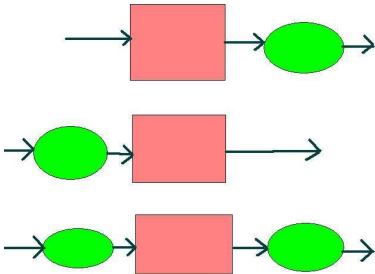
A much used idea is to build up structures from simple building blocks. This could correspond both to physical insights and as a means for generating flexible structures.

Building Blocks:



Basic building blocks for block-oriented models. Square: A linear dynamic system. Oval: A nonlinear static transformation

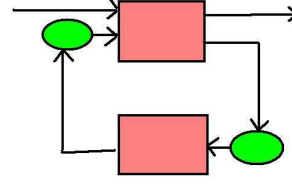
Common Models:



Typical block oriented models. Above: A Wiener model. Middle: A Hammerstein model, Below: A Hammerstein-Wiener model.

These connections may correspond to physical phenomena. The Wiener model is a linear system followed by nonlinear sensors and the Hammerstein model has nonlinear actuators. Both these cases are common in practice. One may also note that the Wiener model, if allowed to have multiple linear outputs becomes a universal approximator to a wide class of nonlinear systems, cf (Boyd and Shua, 1985).

Other Combinations: A Wiener-Hammerstein model is the counterpart with two linear dynamic systems connected via a static nonlinearity. It is also possible to define more complex combinations of the blocks, with feedback etc.



Recently, such structured have been found to be useful in several contexts, see (Hsu *et al.*, 2006) and (Schoukens *et al.*, 2003). With the linear blocks parameterized as a linear dynamic system and the static blocks parameterized as a function (“curve”), this gives a parameterization of the output as

$$\hat{y}(t|\theta) = g(Z^{t-1}, \theta)$$

and the general approach of parametric model fitting can be applied.

However, in this contexts many algorithmic variants have been suggested, especially to initialize the search, e.g., (Bai, 2002)

C. Composite Local Models

Nonlinear systems are often handled by linearization around a working point.

The idea behind *composite local models* is to deal with the nonlinearities by developing local models, which are good approximations in different neighborhoods, and then compose a global model from these. Often, the local models are linear, so a common name for composite models is also *local linear models*. See, e.g. (Johansen and Foss, 1995), and (Murray-Smith and Johansen, 1997).

The concept is best illustrated by a simple example: Consider a tank with inflow u and outflow y and level h : The dynamics is described by the following equations:

$$\begin{aligned} \dot{h} &= -\sqrt{h} + u \\ y &= \sqrt{h} \end{aligned}$$

Linearize around level h^* with corresponding flows $u^* = y^* = \sqrt{h^*}$:

$$\begin{aligned} \dot{h} &= -\frac{1}{2\sqrt{h^*}}(h - h^*) + (u - u^*) \\ y &= y^* + \frac{1}{2\sqrt{h^*}}(h - h^*) \end{aligned}$$

Sample this linearized model with sampling time T_s to obtain a one-step ahead prediction of the output:

$$\begin{aligned} \hat{y}_{h^*}(t) &= \theta_{h^*}^T \varphi(t) \\ \varphi(t) &= [1 \quad -y(t - T_s) \quad u(t - T_s)]^T \\ \theta_{h^*} &= [\gamma_{h^*} \quad \alpha_{h^*} \quad \beta_{h^*}]^T \end{aligned}$$

where α, β, γ are numerical values that depend on the level h^* . To form a total, composite model, select or average over

these local predictions, computed at a grid of values of h^*

$$\hat{y}(t) = \sum_{k=1}^d w_k(h, h_k) \hat{y}_{h_k}(t)$$

The choice of weights w_k is similar to (15). One choice could be that only one w_k is non-zero, thus selecting the local model that is closest to the actual value of h .

General Comments.: Let the measured working point variable (tank level h in the example) be denoted by $\rho(t)$ (sometimes called *regime variable*). If the regime variable is partitioned into d values ρ_k , the predicted output will be

$$\hat{y}(t) = \sum_{k=1}^d w_k(\rho(t), \rho_k) \hat{y}^{(k)}(t)$$

The prediction $\hat{y}^{(k)}(t)$ is the local model corresponding to ρ_k . This prediction depends on some parameters that are associated with the k :th local model, which we denote by $\theta^{(k)}$. (The vector θ will contain the parameters of all local models.) If this model is linear in the parameters, $\hat{y}^{(k)}(t) = \varphi^T(t)\theta^{(k)}$ the whole model will be a linear regression in the parameters θ .

Building a Composite Local Model.: To build the model, we need to

- Select the regime variable ρ
- Decide the partition of the regime variable $w_k(\rho(t), \eta)$. Here η is a parameter that describes the partition
- Find the local models in each partition.

If the local models are linear regressions, the total model will be

$$\hat{y}(t, \theta, \eta) = \sum_{k=1}^d w_k(\rho(t), \eta) \varphi^T(t) \theta^{(k)} \quad (27)$$

which for fixed η is a linear regression.

D. Hybrid Models and LPV Models

The model (27) is also an example of a *hybrid* model. It is piecewise linear (or affine), and switches between different modes as the “state” $\varphi(t)$ varies over the partition. The regime variable ρ is then a known function of φ . If the partition is given, so that η is known, the estimation problem is simple: It is a linear regression. However, if the partition has to be estimated too, the problem is considerably more difficult, due to the discrete/logical nature of the influence of η . Methods based on mixed integer and linear (or quadratic) programming are described in (Roll *et al.*, 2004) and (Bemporad *et al.*, 2003).

So called *Linear Parameter Varying (LPV)* models are also closely related to composite local models. In state space form they are described by:

$$\begin{aligned} \dot{x}(t) &= A(\rho(t))x(t) + B(\rho(t))u(t) \\ y(t) &= C(\rho(t))x(t) + D(\rho(t))u(t) \end{aligned}$$

where the *exogenous* or regime parameter $\rho(t)$ is measured during the operation of the system. Identification of such models have been the subject of recent interest. See, e.g., (Lee and Poolla, 1999) and (Bamieh and Giarré, 2002).

VIII. MAKING THE TECHNIQUES AVAILABLE AND ACCESSIBLE

The field of non-linear system identification is, as stressed several times here, an extensive and versatile area. It is easy to get confused by the vast number of approaches and variants of methods available. Therefore it is especially important to package (a subset of) the possible identification tools in a user-friendly way. It is clearly a special challenge to design the syntax so that the complex theory may be accessible to users without having them exposed to all the intricate choices. An attempt to do that will be contained in the next version of the MathWorks System Identification Toolbox (SITB), (Ljung, 2003) which will integrate techniques for non-linear and linear models. This is a joint project with several contributors, as outlined in (Ljung *et al.*, 2006).

The basic idea is to do this integration in a transparent manner, so that the rather complex problem of estimating and analyzing non-linear models will appear simple and natural. It is also desirable that it can be done with a syntax which has the same look and feel as for linear models.

The non-nonlinear model structures supported by the toolbox are

- `idnlgrey`:

(Cf Section VI)

Grey-box models corresponding to arbitrary non-linear state-space equations in continuous or discrete time. The user supplies code in the form of MATLAB m-file or C-mex file that defines the right hand side of these state-space equations.

grey-box models corresponding to arbitrary explicit non-linear state-space equations in continuous or discrete time. The user supplies the model structure in the form of a MATLAB m-file or C-mex file that defines the right hand side of the state-space equations.

- `idnlarx`:

Non-linear ARX models: (cf. Section V-D) The system output is modeled as a nonlinear regression of past inputs and past outputs.

- `idnlhw`:

these are non-linear block-oriented models of Hammerstein-Wiener type (Section VII-B).

Like the linear models of the SITB, each nonlinear model is implemented as a MATLAB object. A model thus has a certain number of properties, like all MATLAB objects. Once a model is created, its properties can be accessed by the commands `get/set` following the standard syntax. Property/Value pairs can also be used in other method functions. For example, when a nonlinear model object `m0` has been created, the estimation syntax is the same as for linear models:

```
m = pem(data, m0, P1, V1, . . . , Pn, Vn)
```

where `Pk, Vk` are optional Property/Value pairs, and `data` is an `iddata` object.

The quality of the estimated models `m1, m2, . . . , mn` can also be evaluated similarly to the linear case by commands like

```
compare(data, m1, m2, . . . , mn)
```

```
resid(data,m1)
sim(data,m1)
predict(data,m1)
```

for model simulation error comparison, residual analysis, simulation and prediction, respectively. The nonlinearities in estimated models are basically plotted by

```
plot(m1,m2,...,mn)
```

For an `idnlgrey` model, the model structure must be specified in a `m-file` or `C-mex` file before being estimated with data.

For `idnlarx` and `idnlhw` models, a simple command can be used to specify a model structure and to estimate the model with data. For example:

```
m = nlarx(data,[2 2 1], 'sigmoidnet')
m = nlwh(data,[2 2 1], 'sigmoidnet',...
         'wavenet')
```

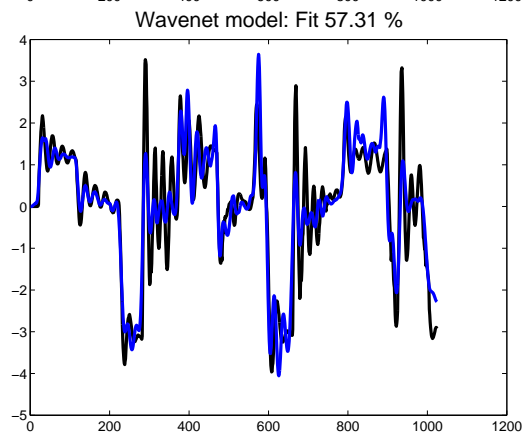
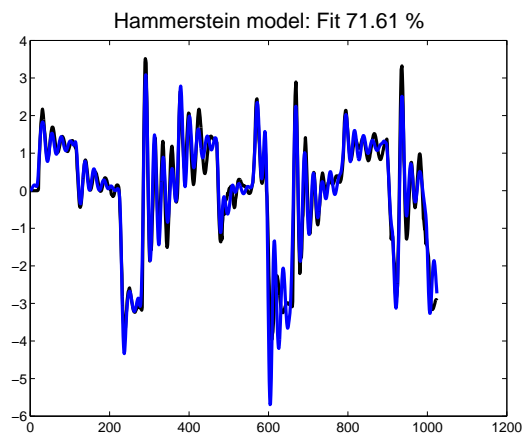
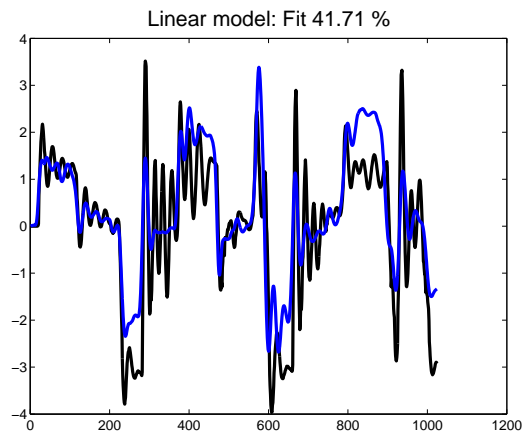
where `'sigmoidnet'` and `'wavenet'` indicate the types of nonlinearity estimators.

IX. EXAMPLE

In this example we load data from a hydraulic crane (forest machine to lift logs). The input is the pressure in the cylinder and the output is the position of the crane tip. First a linear model is estimated based on the first half of the data. The simulated output is then compared to the measured output for the whole data record. After that a Hammerstein model is tried out, with a sigmoidal neural net with 10 neurons as the input, static nonlinearity. The third model is an `nlarx` model with a Wavenet nonlinearity affecting only the past inputs (the past outputs entering linearly).

The figures show the measured output as a thick line, and the model simulated output as a thin line. They also show the fit of the model (in terms of the percentage of the measured output variation that is reproduced by the model). Clearly, the Hammerstein model gives the best performance in this case. The physical explanation for this could be that the measured hydraulic cylinder pressure is transformed to forces acting on the mechanical parts by a non-linear static function, whereafter the dynamics is described by linear mechanical motion equations.

```
load robotarm
data = iddata(y,u);
date=data(1:512);
datv=data(513:end);
m1 = arx(date,[3 2 1]);
compare(data,m1);
mh=nlhw(date,[2 3 1], 'sig', unitgain);
compare(data,mh);
mw=nlarx(date,[3 2 1], 'wave', 'nlreg', [4 5]);
compare(data,mw);
```



X. CONCLUSIONS

Identification of nonlinear models is indeed a problem with many facets. What makes the area so overwhelmingly rich is that so many different research communities have contributed and continue to contribute. Many issues are purely statistical and do not depend on whether the underlying process is a dynamical system or not. For the control community it is worth while to find its niche with the best chances to give valuable contributions. A few suggestions are

- Find a working relationship between modern physical modeling tools and the estimation of dynamical models. Deal directly with Differential Algebraic Equations, and sort out how to work with disturbance descriptions in such models.
- Study identifiability questions when sub-models are connected in such object oriented modeling environments.


- Consider both algebraic tools (like Ritt’s algorithm) and algorithmic tricks to convexify estimation problems in order to provide powerful initialization steps for parameter estimation. Can particle filters and the EM algorithm offer help in dealing with (i.e. avoiding) local optima of the likelihood function?
- Can tools of considerable generality be developed from block-oriented models (Section VII-B)?
- Can black-box models be developed that allow a better handle on stability for simulation (cf. Section V-C)?
- Since linear dynamic models will remain the basic arena for control applications, it is important to fully understand how linear models approximate real-life nonlinear systems. Cf. (Enqvist, 2005).

XI. ACKNOWLEDGMENTS

This work was supported by the Swedish Research Council, which is gratefully acknowledged. I also thank my students for teaching me important stuff in the area.

REFERENCES

- Andrieu, C., A. Doucet, S. S. Singh and V. B. Tadić (2004). Particle methods for change detection, system identification and control. *Proceeding of IEEE* **92**(3), 423–438.
- Bai, E. W. (2002). A blind approach to the Hammerstein-Wiener model identification. *Automatica* **38**(6), 967–979.
- Bamieh, B. and L. Giarré (2002). Identification of linear parameter varying models. *Int. Journal of Robust and Nonlinear Control* **12**, 841–853.
- Baron, A.R. (1989). Statistical properties of artificial neural networks. In: *Proceedings of the 28th IEEE Conference on Decision and Control*. pp. 280–285.
- Bemporad, A., A. Garulli, S. Paoletti and A. Vicino (2003). A greedy approach to identification of piecewise affine models. In: *Hybrid Systems, Computation and Control* (O. Maler and A. Pnueli, Eds.). pp. 97–112. Number 2623 In: *Lecture Notes in Computer Science*. Springer Verlag.
- Billings, S. A. (1990). Identification of nonlinear systems - a survey. *IEE Proc. D* **127**, 272–285.
- Bohlin, T. (2006). *Practical Grey-box Process Identification*. Springer-Verlag, London.
- Boyd, S. and L. O. Shua (1985). Fading memory and the problem of approximating nonlinear operators with volterra series. *IEEE Transactions on Circuits and Systems* **CAS-32**(11), 1150–1161.
- Enqvist, M. (2005). Linear Models of Nonlinear Systems. PhD thesis. Linköping University, Sweden. Linköping Studies in Science and Technology. Dissertation No 985.
- Fan, J. and I. Gijbels (1996). *Local Polynomial Modelling and Its Applications*. number 66 In: *Monographs on Statistics and Applied Probability*. Chapman & Hall.
- Fritzson, P. (2004). *Object-oriented Modeling and Simulation with MODELICA 2.1.1*. IEEE Press. Piscataway, NJ.
- Fukunaga, K. (1990). *Introduction to statistical pattern recognition (2nd ed.)*. Academic Press, New York.
- Gerdin, M. (2006). Identification of Dynamic Models Described by Differential Algebraic Equations. PhD thesis. Department of Electrical Engineering, Linköping University, Linköping, Sweden.
- Golub, G. H., M. Heath and G. Wahba (1979). Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics* **21**(2), 215–223.
- Harris, C., X. Hong and Q. Gan (2002). *Adaptive Modelling, Estimation and Fusion from Data: A Neurofuzzy Approach*. Springer. New York.
- Hastie, T., R. Tibshirani and J. Friedman (2001). *The Elements of Statistical Learning*. Springer.
- Härdle, W. (1990). *Applied Nonparametric Regression*. Cambridge University Press. Cambridge, UK.
- Hsu, K., T. Vincent and K. Poolla (2006). A kernel based approach to structured nonlinear system identification part I: Algorithms, part II: Convergence and consistency. In: *Proc. IFAC Symposium on System Identification*. Newcastle, Australia.
- Jang, J.-S. R. and C.-T. Sun (1995). Neuro-fuzzy modeling and control. *Proc. of the IEEE* **83**(3), 378–406.
- Johansen, T. A. and B. A. Foss (1995). Identification of nonlinear-system structure and parameters using regime decomposition. *Automatica* **31**(2), 321–326.
- Lee, L. and K. Poolla (1999). Identification of linear parameter-varying systems using non-linear programming. *ASME Journal of Dynamic Systems, Measurement and Control* **121**, 71–78.
- Ljung, L. (1999). *System Identification - Theory for the User*. 2nd ed.. Prentice-Hall. Upper Saddle River, N.J.
- Ljung, L. (2003). *The System Identification Toolbox: The Manual*. The MathWorks Inc. 1st edition 1986, 6th edition 2003. Natick, MA, USA.
- Ljung, L. (2006). Some aspects of nonlinear system identification. In: *Proc. 14th IFAC Symposium on System Identification*. Newcastle, Australia.
- Ljung, L. and T. Glad (1994). On global identifiability of arbitrary model parameterizations. *Automatica* **30**(2), pp 265–276.
- Ljung, L., Q. Zhang, P. Lindskog, A. Juditsky and R. Singh (2006). An integrated system identification toolbox for linear and non-linear models. In: *Proc. 14th IFAC Symposium on System Identification*. Newcastle, Australia.
- Murray-Smith, R. and Johansen, T. A., Eds.) (1997). *Multiple Model Approaches to Modeling and Control*. Taylor and Francis. London.
- Nadaraya, E. (1964). On estimating regression. *Theory of Prob. and Applic.* **9**, 141–142.
- Nelles, O. (2001). *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*. Springer Verlag. Berlin.
- Roll, J., A. Bemporad and L. Ljung (2004). Identification of piecewise affine systems via mixed-integer programming. *Automatica* **40**(1), 37–50.
- Roll, J., A. Nazin and L. Ljung (2005). Non-linear system identification via direct weight optimization. *Automatica* **41**(3), 475–490.
- Sacks, J. and D. Ylvisaker (1978). Linear estimation for approximately linear models. *The Annals of Statistics* **6**(5), 1122–1137.
- Schittkowski, K. (2002). *Numerical Data Fitting in Dynamical Systems*. Kluwer Academic Publishers. Dordrecht.
- Schön, T. B., A. Wills and B. Ninness (2006). Maximum likelihood nonlinear system estimation. In: *Proceedings of the 14th IFAC Symposium on System Identification*. Newcastle, Australia. Accepted for publication.
- Schoukens, J., J. Nemeth, P. Crama, Y. Rolain and R. Pintelon (2003). Fast approximate identification of nonlinear systems. In: *Proc. 13th IFAC Symposium on System Identification* (P. van der Hof, B. Wahlberg and S. Weiland, Eds.). Rotterdam, The Netherlands. pp. 61–66.
- 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.
- Suykens, J.A.K., T. van Gestel, J. De Brabanter, B. De Moor and J. Vandewalle (2002). *Least Squares Support Vector Machines*. World Scientific. Singapore.
- Vapnik, V. (1998). *Statistical Learning Theory*. Wiley.
- Zhang, Q. and A. Benveniste (1992). Wavelet networks. *IEEE Trans Neural Networks* **3**, 889–898.

	Avdelning, Institution Division, Department Division of Automatic Control Department of Electrical Engineering	Datum Date 2007-06-14
	Språk Language <input type="checkbox"/> Svenska/Swedish <input checked="" type="checkbox"/> Engelska/English <input type="checkbox"/> _____	Rapporttyp Report category <input type="checkbox"/> Licentiatavhandling <input type="checkbox"/> Examensarbete <input type="checkbox"/> C-uppsats <input type="checkbox"/> D-uppsats <input checked="" type="checkbox"/> Övrig rapport <input type="checkbox"/> _____
URL för elektronisk version http://www.control.isy.liu.se		LiTH-ISY-R-2784
Titel Identification of Nonlinear Systems Title		
Författare Lennart Ljung Author		
Sammanfattning Abstract Identification of nonlinear systems is a problem with many facets and roots in several diverse fields. It is not possible to survey the area in a short text. The current presentation gives a subjective view on some essential features in the area. These concern a classification of methods, the use of physical insight in models, and some overall issues like bias-variance trade-off. It is also discusses how the methods can be made available in software packages.		
Nyckelord Keywords identification, non-linear systems		