# System Identification

Lennart Ljung
Department of Electrical Engineering, Linköping University
S-581 83 Linköping, Sweden. e-mail `ljung@isy.liu.se`

May 29, 1995

# 1 The Basic Ideas

## 1.1 10 Basic Questions About System Identification

1. **What is System Identification?**
   System Identification allows you to build mathematical models of a dynamic system based on measured data.

2. **How is that done?**
   Essentially by adjusting parameters within a given model until its output coincides as well as possible with the measured output.

3. **How do you know if the model is any good?**
   A good test is to take a close look at the model's output compared to the measurements on a data set that wasn't used for the fit ("Validation Data").

4. **Can the quality of the model be tested in other ways?**
   It is also valuable to look at what the model couldn't reproduce in the data ("the residuals"). There should be no correlation with other available information, such as the system's input.

5. **What models are most common?**
The techniques apply to very general models. Most common models are difference equations descriptions, such as ARX and ARMAX models, as well as all types of linear state-space models. Lately, black-box non-linear structures, such as Artifical Neural Networks, Fuzzy models, and so on, have been much used.

6. **Do you have to assume a model of a particular type?**
For parametric models, you have to specify the structure. However, if you just assume that the system is linear, you can directly estimate its impulse or step response using Correlation Analysis or its frequency response using Spectral Analysis. This allows useful comparisons with other estimated models.

7. **How do you know what model structure to try?**
Well, you don't. For real life systems there is never any "true model", anyway. You have to be generous at this point, and try out several different structures.

8. **Can non-linear models be used in an easy fashion?**
Yes. Most common model nonlinearities are such that the measured data should be nonlinearly transformed (like squaring a voltage input if you think that it's the power that is the stimulus). Use physical insight about the system you are modeling and try out such transformations on models that are linear in the new variables, and you will cover a lot.

9. **What does this article cover?**
After reviewing an archetypical situation in this section, we describe the basic techniques for parameter estimation in arbitrary model structures. Section 3 deals with linear models of black-box structure, and Section 4 deals with particular estimation methods that can be used (in addition to the general ones) for such models. Physically parameterized model structures are described in Section 5, and non-linear black box models (including neural networks) are discussed in Section 6. The final Section 7 deals with the choices and decisions the user is faced with.

10. **Is this really all there is to System Identification?**
Actually, there is a huge amount written on the subject. Experience

with real data is the driving force to further understanding. It is important to remember that any estimated model, no matter how good it looks on your screen, has only picked up a simple reflection of reality. Surprisingly often, however, this is sufficient for rational decision making.

## 1.2 Background and Literature

System Identification has its roots in standard statistical techniques and many of the basic routines have direct interpretations as well known statistical methods such as Least Squares and Maximum Likelihood. The control community took an active part in the development and application of these basic techniques to dynamic systems right after the birth of "modern control theory" in the early 1960's. Maximum likelihood estimation was applied to difference equations (ARMAX models) by [Åström and Bohlin, 1965] and thereafter a wide range of estimation techniques and model parameterizations flourished. By now, the area is well matured with established and well understood techniques. Industrial use and application of the techniques has become standard. See [Ljung, 1986] for a common software package.

The literature on System Identification is extensive. For a practical user oriented introduction we may mention [Ljung and Glad, 1994]. Texts that go deeper into the theory and algorithms include [Ljung, 1987], and [Söderström and Stoica, 1989]. A classical treatment is [Box and Jenkins, 1970].

These books all deal with the "mainstream" approach to system identification, as described in this article. In addition, there is a substantial literature on other approaches, such as "set membership" (compute all those models that reproduce the observed data within a certain given error bound), estimation of models from given frequency response measurement [Schoukens and Pintelon, 1991], on-line model estimation [Ljung and Söderström, 1983], non-parametric frequency domain methods [Brillinger, 1981], etc. To follow the development in the field, the IFAC series of Symposia on System Identification (Budapest, 1991, Copenhagen, 1994) is a good source.

## 1.3 An Archetypical Problem – ARX Models and the Linear Least Squares Method

### The Model

We shall generally denote the system's input and output at time $t$ by $u(t)$ and $y(t)$, respectively. Perhaps the most basic relationship between the input and output is the *linear difference equation*

$$y(t) + a_1 y(t-1) + \ldots + a_n y(t-n) = b_1 u(t-1) + \ldots + b_m u(t-m) \quad (1)$$

We have chosen to represent the system in *discrete time*, primarily since observed data are always collected by sampling. It is thus more straightforward to relate observed data to discrete time models. Nothing prevents us however from working with continuous time models: we shall return to that in Section 5.

In (1) we assume the *sampling interval* to be one time unit. This is not essential, but makes notation easier.

A pragmatic and useful way to see (1) is to view it as a way of *determining the next output value* given previous observations:

$$y(t) = -a_1 y(t-1) - \ldots - a_n y(t-n) + b_1 u(t-1) + \ldots + b_m u(t-m) \quad (2)$$

For more compact notation we introduce the vectors

$$\theta = [a_1, \ldots, a_n \, b_1, \ldots, b_m]^T \quad (3)$$

$$\varphi(t) = [-y(t-1) \ldots - y(t-n) \; u(t-1) \ldots u(t-m)]^T \quad (4)$$

With these (2) can be rewritten as

$$y(t) = \varphi^T(t)\theta$$

To emphasize that the calculation of $y(t)$ from past data (2) indeed depends on the parameters in $\theta$, we shall rather call this calculated value $\hat{y}(t|\theta)$ and write

$$\hat{y}(t|\theta) = \varphi^T(t)\theta \tag{5}$$

**The Least Squares Method**

Now suppose for a given system that we do not know the values of the parameters in $\theta$, but that we have recorded inputs and outputs over a time interval $1 \leq t \leq N$:

$$Z^N = \{u(1), y(1), \ldots, u(N), y(N)\} \tag{6}$$

An obvious approach is then to select $\theta$ in (1) through (5) so as to fit the calculated values $\hat{y}(t|\theta)$ as well as possible to the measured outputs by the least squares method:

$$\min_\theta V_N(\theta, Z^N) \tag{7}$$

where

$$\begin{aligned} V_N(\theta, Z^N) &= \frac{1}{N}\sum_{t=1}^{N}(y(t) - \hat{y}(t|\theta))^2 = \\ &= \frac{1}{N}\sum_{t=1}^{N}(y(t) - \varphi^T(t)\theta)^2 \end{aligned} \tag{8}$$

We shall denote the value of $\theta$ that minimizes (7) by $\hat{\theta}_N$:

$$\hat{\theta}_N = \arg\min_\theta V_N(\theta, Z^N) \tag{9}$$

("arg min" means the minimizing argument, i.e., that value of $\theta$ which minimizes $V_N$.)

Since $V_N$ is quadratic in $\theta$, we can find the minimum value easily by setting the derivative to zero:

$$0 = \frac{d}{d\theta} V_N(\theta, Z^N) = \frac{2}{N} \sum_{t=1}^{N} \varphi(t)(y(t) - \varphi^T(t)\theta)$$

which gives

$$\sum_{t=1}^{N} \varphi(t) y(t) = \sum_{t=1}^{N} \varphi(t)\varphi^T(t)\theta \tag{10}$$

or

$$\hat{\theta}_N = \left[\sum_{t=1}^{N} \varphi(t)\varphi^T(t)\right]^{-1} \sum_{t=1}^{N} \varphi(t) y(t) \tag{11}$$

Once the vectors $\varphi(t)$ are defined, the solution can easily be found by modern numerical software, such as MATLAB.

**Example 1** *First order difference equation*

*Consider the simple model*

$$y(t) + ay(t-1) = bu(t-1).$$

*This gives us the estimate according to (3), (4) and (11)*

$$\begin{bmatrix} \hat{a}_N \\ \hat{b}_N \end{bmatrix} = \begin{bmatrix} \sum y^2(t-1) & -\sum y(t-1)u(t-1) \\ -\sum y(t-1)u(t-1) & \sum u^2(t-1) \end{bmatrix}^{-1} \begin{bmatrix} -\sum y(t)y(t-1) \\ \sum y(t)u(t-1) \end{bmatrix}$$

*All sums are from $t = 1$ to $t = N$. A typical convention is to take values outside the measured range to be zero. In this case we would thus take $y(0) = 0$.*

The simple model (1) and the well known least squares method (11) form the archetype of System Identification. Not only that – they also give the most commonly used parametric identification method and are much more versatile than perhaps perceived at first sight. In particular one should realize that (1) can directly be extended to several different inputs (this just calls for a redefinition of $\varphi(t)$ in (4)) and that the inputs and outputs do not have to be the raw measurements. On the contrary – it is often most important to think over the physics of the application and come up with suitable inputs and outputs for (1), formed from the actual measurements.

**Example 2** An immersion heater

*Consider a process consisting of an immersion heater immersed in a cooling liquid. We measure:*

- $v(t)$*: The voltage applied to the heater*
- $r(t)$*: The temperature of the liquid*
- $y(t)$*: The temperature of the heater coil surface*

*Suppose we need a model for how* $y(t)$ *depends on* $r(t)$ *and* $v(t)$*. Some simple considerations based on common sense and high school physics ("Semi-physical modeling") reveal the following:*

- *The change in temperature of the heater coil over one sample is proportional to the electrical power in it (the inflow power) minus the heat loss to the liquid*
- *The electrical power is proportional to* $v^2(t)$
- *The heat loss is proportional to* $y(t) - r(t)$

*This suggests the model*

$$y(t) = y(t-1) + \alpha v^2(t-1) - \beta(y(t-1) - r(t-1))$$

*which fits into the form*

$$y(t) + \theta_1 y(t-1) = \theta_2 v^2(t-1) + \theta_3 r(t-1))$$

*This is a two input ($v^2$ and $r$) and one output model, and corresponds to choosing*

$$\varphi(t) = [-y(t-1) \quad v^2(t-1) \quad r(t-1)]^T$$

*in (5).*

## Some Statistical Remarks

Model structures, such as (5) that are linear in $\theta$ are known in statistics as *linear regression* and the vector $\varphi(t)$ is called the *regression vector* (its components are the *regressors*). "Regress" here alludes to the fact that we try to calculate (or describe) $y(t)$ by "going back" to $\varphi(t)$. Models such as (1) where the regression vector – $\varphi(t)$ – contains old values of the variable to be explained – $y(t)$ – are then partly *auto-regressions*. For that reason the model structure (1) has the standard name ARX-model (Auto-regression with extra inputs).

There is a rich statistical literature on the properties of the estimate $\hat{\theta}_N$ under varying assumptions. See, e.g. [Draper and Smith, 1981]. So far we have just viewed (7) and (8) as "curve-fitting". In Section 2.2 we shall deal with a more comprehensive statistical discussion, which includes the ARX model as a special case. Some direct calculations will be done in the following subsection.

## Model Quality and Experiment Design

Let us consider the simplest special case, that of a Finite Impulse Response (FIR) model. That is obtained from (1) by taking $n = 0$:

$$y(t) = b_1 u(t-1) + \ldots b_m u(t-m) \qquad (12)$$

Suppose that the observed data really have been generated by a similar mechanism

$$y(t) = b_1^0 u(t-1) + \ldots b_m^0 u(t-m) + e(t) \tag{13}$$

where $e(t)$ is a white noise sequence with variance $\lambda$, but otherwise unknown. (That is, $e(t)$ can be described as a sequence of independent random variables with zero mean values and variances $\lambda$.) Analogous to (5), we can write this as

$$y(t) = \varphi^T(t)\theta_0 + e(t) \tag{14}$$

We can now replace $y(t)$ in (11) by the above expression, and obtain

$$\begin{aligned} \hat{\theta}_N &= \left[\sum_{t=1}^{N} \varphi(t)\varphi^T(t)\right]^{-1} \sum_{t=1}^{N} \varphi(t)y(t) \\ &= \left[\sum_{t=1}^{N} \varphi(t)\varphi^T(t)\right]^{-1} \left[\sum_{t=1}^{N} \varphi(t)\varphi^T(t)\theta_0 + \sum_{t=1}^{N} \varphi(t)e(t)\right] \end{aligned}$$

or

$$\tilde{\theta}_N = \hat{\theta}_N - \theta_0 = \left[\sum_{t=1}^{N} \varphi(t)\varphi^T(t)\right]^{-1} \sum_{t=1}^{N} \varphi(t)e(t) \tag{15}$$

Suppose that the input $u$ is independent of the noise $e$. Then $\varphi$ and $e$ are independent in this expression, so it is easy to see that $E\tilde{\theta}_N = 0$, since $e$ has zero mean. The estimate is consequently *unbiased*. Here $E$ denotes *mathematical expectation*.

We can also form the expectation of $\tilde{\theta}_N \tilde{\theta}_N^T$, i.e., the covariance matrix of the parameter error. Denote the matrix within brackets by $R_N$. Take expectation with respect to the white noise $e$. Then $R_N$ is a deterministic matrix and we have

$$P_N = E\tilde{\theta}_N \tilde{\theta}_N^T = R_N^{-1} \sum_{t,s=1}^{N} \varphi(t)\varphi^T(s) Ee(t)e(s) R_N^{-1} = \lambda R_N^{-1} \tag{16}$$

since the double sum collapses to $\lambda R_N$.

We have thus computed the covariance matrix of the estimate $\hat{\theta}_N$. It is determined entirely by the input properties and the noise level. Moreover define

$$\bar{R} = \lim_{N\to\infty} \frac{1}{N} R_N \tag{17}$$

This will be the *covariance matrix* of the input, i.e. the $i-j$-element of $\bar{R}$ is $R_{uu}(i-j)$, as defined by (89) later on.

If the matrix $\bar{R}$ is non-singular, we find that the covariance matrix of the parameter estimate is approximately (and the approximation improves as $N \to \infty$)

$$P_N = \frac{\lambda}{N} \bar{R}^{-1} \tag{18}$$

A number of things follow from this. All of them are typical of the general properties to be described in Section 2.2:

- The covariance decays like $1/N$, so the parameters approach the limiting value at the rate $1/\sqrt{N}$.
- The covariance is proportional to the Noise-To-Signal ratio. That is, it is proportional to the noise variance and inversely proportional to the input power.
- The covariance does not depend on the input's or noise's signal shapes, only on their variance/covariance properties.
- Experiment design, i.e., the selection of the input $u$, aims at making the matrix $\bar{R}^{-1}$ "as small as possible". Note that the same $\bar{R}$ can be obtained for many different signals $u$.

## 1.4 The Main Ingredients

The main ingredients for the System Identification problem are as follows

- The data set $Z^N$
- A class of candidate model descriptions; *a Model Structure.*
- A criterion of fit between data and models.
- Routines to validate and accept resulting models.

We have seen in Section 1.3 a particular model structure, the ARX-model. In fact the major problem in system identification is to select a good model structure, and a substantial part of this article deals with various model structures. See Sections 3, 5, and 6, which all concern this problem. Generally speaking, a model structure is a parameterized mapping from past inputs and outputs $Z^{t-1}$ (cf (6)) to the space of the model outputs:

$$\hat{y}(t|\theta) = g(\theta, Z^{t-1}) \tag{19}$$

Here $\theta$ is the finite dimensional vector used to parameterize the mapping.

Actually, the problem of fitting a given model structure to measured data is much simpler, and can be dealt with independently of the model structure used. We shall do so in the following section.

The problem of assuring a data set with adequate information contents is the problem of *experiment design*, and it will be described in Section 7.1.

Model validation is both a process to discriminate between various model structures and the final quality control station, before a model is delivered to the user. This problem is discussed in Section 7.2.

# 2 General Parameter Estimation Techniques

In this section we shall deal with issues that are independent of model structure. Principles and algorithms for fitting models to data, as well as the general properties of the estimated models are all model-structure independent and equally well applicable to, say, ARMAX models and Neural Network models.

The section is organized as follows. In Section 2.1 the general principles for parameter estimation are outlined. Sections 2.2 and 2.3 deal with the asymptotic (in the number of observed data) properties of the models, while algorithms, both for on-line and off-line use are described in Section 2.5.

## 2.1 Fitting Models to Data

In Section 1.3 we showed one way to parameterize descriptions of dynamical systems. There are many other possibilities and we shall spend a fair amount of this contribution to discuss the different choices and approaches. *This is actually the key problem in system identification.* No matter how the problem is approached, the bottom line is that such a model parameterization leads to a predictor

$$\hat{y}(t|\theta) = g(\theta, Z^{t-1}) \tag{20}$$

that depends on the unknown parameter vector and past data $Z^{t-1}$ (see (6). This predictor can be linear in $y$ and $u$. This in turn contains several special cases both in terms of black-box models and physically parameterized ones, as will be discussed in Sections 3 and 5, respectively. The predictor could also be of general, non-linear nature, as will be discussed in Section 6.

In any case *we now need a method to determine a good value of $\theta$,* based on the information in an observed, sampled data set (6). It suggests itself that the basic least-squares like approach (7) through (9) still is a natural approach, even when the predictor $\hat{y}(t|\theta)$ is a more general function of $\theta$.

A procedure with some more degrees of freedom is the following one

1. From observed data and the predictor $\hat{y}(t|\theta)$ form the sequence of prediction errors,

$$\varepsilon(t, \theta) = y(t) - \hat{y}(t|\theta), \quad t = 1, 2, \ldots N \tag{21}$$

2. Possibly filter the prediction errors through a linear filter $L(q)$,

$$\varepsilon_F(t,\theta) = L(q)\varepsilon(t,\theta) \tag{22}$$

(here $q$ denotes the shift operator, $qu(t) = u(t+1)$) so as to enhance or depress interesting or unimportant frequency bands in the signals.

3. Choose a scalar valued, positive function $\ell(\cdot)$ so as to measure the "size" or "norm" of the prediction error:

$$\ell(\varepsilon_F(t,\theta)) \tag{23}$$

4. Minimize the sum of these norms:

$$\hat{\theta}_N = \arg\min_\theta V_N(\theta, Z^N) \tag{24}$$

where

$$V_N(\theta, Z^N) = \frac{1}{N}\sum_{t=1}^{N} \ell(\varepsilon_F(t,\theta)) \tag{25}$$

This procedure is natural and pragmatic – we can still think of it as "curve-fitting" between $y(t)$ and $\hat{y}(t|\theta)$. It also has several statistical and information theoretic interpretations. Most importantly, if the noise source in the system (like in (62) below) is supposed to be a sequence of independent random variables $\{e(t)\}$ each having a probability density function $f_e(x)$, then (24) becomes the Maximum Likelihood estimate (MLE) if we choose

$$L(q) = 1 \quad \text{and} \quad \ell(\varepsilon) = -\log f_e(\varepsilon) \tag{26}$$

The MLE has several nice statistical features and thus gives a strong "moral support" for using the outlined method. Another pleasing aspect is that the method is independent of the particular model parameterization used (although this will affect the actual minimization procedure). For example, the method of "back propagation" often used in connection with neural network parameterizations amounts to computing $\hat{\theta}_N$ in (24) by a recursive gradient method. We shall deal with these aspects in Section 2.5.

## 2.2 Model Quality

An essential question is, of course, what properties will the estimate resulting from (24) have. These will naturally depend on the properties of the data record $Z^N$ defined by (6). It is in general a difficult problem to characterize the quality of $\hat{\theta}_N$ exactly. One normally has to be content with the asymptotic properties of $\hat{\theta}_N$ as the number of data, $N$, tends to infinity.

It is an important aspect of the general identification method (24) that the asymptotic properties of the resulting estimate can be expressed in general terms for arbitrary model parameterizations.

The first basic result is the following one:

$$\hat{\theta}_N \to \theta^* \quad \text{as} \quad N \to \infty \quad \text{where} \tag{27}$$

$$\theta^* = \arg\min_{\theta} E\ell(\varepsilon_F(t,\theta)) \tag{28}$$

That is, as more and more data become available, the estimate converges to that value $\theta^*$, that would minimize the expected value of the "norm" of the filtered prediction errors. This is in a sense *the best possible approximation* of the true system that is available within the model structure. The expectation $E$ in (28) is taken with respect to all random disturbances that affect the data and it also includes averaging over the input properties. This means in particular that $\theta^*$ will make $\hat{y}(t|\theta^*)$ a good approximation of $y(t)$ with respect to those aspects of the system that are enhanced by the input signal used.

The second basic result is the following one: If $\{\varepsilon(t,\theta^*)\}$ is approximately white noise, then the covariance matrix of $\hat{\theta}_N$ is approximately given by

$$E(\hat{\theta}_N - \theta^*)(\hat{\theta}_N - \theta^*)^T \sim \frac{\lambda}{N}[E\psi(t)\psi^T(t)]^{-1} \tag{29}$$

where

$$\lambda = E\varepsilon^2(t,\theta^*) \tag{30}$$

$$\psi(t) = \frac{d}{d\theta}\hat{y}(t|\theta)|_{\theta=\theta^*} \tag{31}$$

Think of $\psi$ as the sensitivity derivative of the predictor with respect to the parameters. Then (29) says that the covariance matrix for $\hat{\theta}_N$ is proportional to the inverse of the covariance matrix of this sensitivity derivative. This is a quite natural result.

**Note:** For all these results, the expectation operator $E$ can, under most general conditions, be replaced by the limit of the sample mean, that is

$$E\psi(t)\psi^T(t) \leftrightarrow \lim_{N\to\infty} \frac{1}{N}\sum_{t=1}^{N} \psi(t)\psi^T(t) \tag{32}$$

$\square$

The results (27) through (31) are general and hold for all model structures, both linear and non-linear ones, subject only to some regularity and smoothness conditions. They are also fairly natural, and will give the guidelines for all user choices involved in the process of identification. See [Ljung, 1987] for more details around this.

## 2.3 Measures of Model Fit

Some quite general expressions for the expected model fit, that are independent of the model structure, can also be developed.

Let us measure the (average) fit between any model (20) and the true system as

$$\bar{V}(\theta) = E|y(t) - \hat{y}(t|\theta)|^2 \tag{33}$$

Here expectation E is over the data properties (i.e. expectation over "$Z^\infty$" with the notation (6)). Recall that expectation also can be interpreted as sample means as in (32).

Before we continue, let us note the very important aspect that the fit $\bar{V}$ will depend, not only on the model and the true system, *but also on data properties*, like input spectra, possible feedback, etc. We shall say that the fit depends on the *experimental conditions.*

The estimated model parameter $\hat{\theta}_N$ is a random variable, because it is constructed from observed data, that can be described as random variables. To evaluate the model fit, we then take the expectation of $\bar{V}(\hat{\theta}_N)$ with respect to the estimation data. That gives our measure

$$F_N = E\bar{V}(\hat{\theta}_N) \tag{34}$$

In general, the measure $F_N$ depends on a number of things:

- The model structure used.
- The number of data points $N$.
- The data properties for which the fit $\bar{V}$ is defined.
- The properties of the data used to estimate $\hat{\theta}_N$.

The rather remarkable fact is that if the two last data properties coincide, then, asymptotically in $N$, (see, e.g., [Ljung, 1987], Chapter 16)

$$F_N \approx \bar{V}_N(\theta^*)(1 + \frac{dim\theta}{N}) \tag{35}$$

Here $\theta^*$ is the value that minimizes the expected criterion (28). The notation $dim\theta$ means the number of estimated parameters. The result also assumes that the criterion function $\ell(\varepsilon) = ||\varepsilon||^2$, and that the model structure is successful in the sense that $\varepsilon_F(t)$ is approximately white noise.

Despite the reservations about the formal validity of (35), it carries a most important conceptual message: If a model is evaluated on a data set with the same properties as the estimation data, then *the fit will not depend on the* *data properties*, and it will depend on the model structure *only in terms of the number of parameters used and of the best fit offered within the structure.*

The expression can be rewritten as follows. Let $\hat{y}_0(t|t-1)$ denote the "true" one step ahead prediction of $y(t)$, and let

$$W(\theta) = E|\hat{y}_0(t|t-1) - \hat{y}(t|\theta)|^2 \tag{36}$$

and let

$$\lambda = E|y(t) - \hat{y}_0(t|t-1)|^2 \tag{37}$$

Then $\lambda$ is the *innovations* variance, i.e., that part of $y(t)$ that cannot be predicted from the past. Moreover $W(\theta^*)$ is the *bias error*, i.e. the discrepancy between the true predictor and the best one available in the model structure. Under the same assumptions as above, (35) can be rewritten as

$$F_N \approx \lambda + W(\theta^*) + \lambda \frac{dim\theta}{N} \tag{38}$$

The three terms constituting the model error then have the following interpretations

- $\lambda$ is the unavoidable error, stemming from the fact that the output cannot be exactly predicted, even with perfect system knowledge.
- $W(\theta^*)$ is the bias error. It depends on the model structure, and on the experimental conditions. It will typically decrease as $dim\theta$ increases.
- The last term is the *variance error.* It is proportional to the number of estimated parameters and inversely proportional to the number of data points. It does not depend on the particular model structure or the experimental conditions.

## 2.4 Model Structure Selection

The most difficult choice for the user is no doubt to find a suitable model structure to fit the data to. This is of course a very application-dependent problem, and it is difficult to give general guidelines. (Still, some general practical advice will be given in Section 7.)

At the heart of the model structure selection process is to handle the trade-off between bias and variance, as formalized by (38). The "best" model structure is the one that minimizes $F_N$, the fit between the model and the data for a *fresh* data set – one that was not used for estimating the model. Most procedures for choosing the model structures are also aiming at finding this best choice.

### Cross Validation

A very natural and pragmatic approach is *Cross Validation.* This means that the available data set is split into two parts, *estimation data*, $Z_{\text{est}}^{N_1}$ that is used to estimate the models:

$$\hat{\theta}_{N_1} = \arg\min V_{N_1}(\theta, Z_{\text{est}}^{N_1}) \tag{39}$$

and *validation data*, $Z_{\text{val}}^{N_2}$ for which the criterion is evaluated:

$$\hat{F}_{N_1} = V_{N_2}(\hat{\theta}_{N_1}, Z_{\text{val}}^{N_2}) \tag{40}$$

Here $V_N$ is the criterion (25). Then $\hat{F}_N$ will be an unbiased estimate of the measure $F_N$, defined by (34), which was discussed at length in the previous section. The procedure would the be to try out a number of model structures, and choose the one that minimizes $\hat{F}_{N_1}$.

Such cross validation techniques to find a good model structure has an immediate intuitive appeal. We simply check if the candidate model is capable of "reproducing" data it hasn't yet seen. If that works well, we have some confidence in the model, regardless of any probabilistic framework that might be imposed. Such techniques are also the most commonly used ones.

A few comments could be added. In the first place, one could use different splits of the original data into estimation and validation data. For example, in statistics, there is a common cross validation technique called "leave one out". This means that the validation data set consists of one data point "at a time", but successively applied to the whole original set. In the second place, the test of the model on the validation data does not have to be in terms of the particular criterion (40). In system identification it is common practice to simulate (or predict several steps ahead) the model using the validation data, and then visually inspect the agreement between measured and simulated (predicted) output.

## Estimating the Variance Contribution – Penalizing the Model Complexity

It is clear that the criterion (40) has to be evaluated on the validation data to be of any use – it would be strictly decreasing as a function of model flexibility if evaluated on the estimation data. In other words, the adverse effect of the dimension of $\theta$ shown in (38) would be missed. There are a number of criteria – often derived from entirely different viewpoints – that try to capture the influence of this variance error term. The two best known ones are *Akaike's Information Theoretic Criterion, AIC*, which has the form (for Gaussian disturbances)

$$\tilde{V}_N(\theta, Z^N) = \left(1 + \frac{2dim\theta}{N}\right) \frac{1}{N} \sum_{t=1}^{N} \varepsilon^2(t, \theta) \tag{41}$$

and *Rissanen's Minimum Description Length Criterion, MDL* in which $dim\theta$ in the expression above is replaced by $\log N dim\theta$. See [Akaike, 1974a] and [Rissanen, 1978].

The criterion $\tilde{V}_N$ is then to be minimized both with respect to $\theta$ and to a family of model structures. The relation to the expression (35) for $F_N$ is obvious.

## 2.5 Algorithmic Aspects

In this section we shall discuss how to achieve the best fit between observed data and the model, i.e. how to carry out the minimization of (24). For simplicity we here assume a quadratic criterion and set the prefilter $L$ to unity:

$$V_N(\theta) = \frac{1}{2N} \sum_{t=1}^{N} |y(t) - \hat{y}(t|\theta)|^2 \tag{42}$$

No analytic solution to this problem is possible unless the model $\hat{y}(t|\theta)$ is linear in $\theta$, so the minimization has to be done by some numerical search procedure. A classical treatment of the problem of how to minimize the sum of squares is given in [Dennis and Schnabel, 1983].

Most efficient search routines are based on iterative local search in a "downhill" direction from the current point. We then have an iterative scheme of the following kind

$$\hat{\theta}^{(i+1)} = \hat{\theta}^{(i)} - \mu_i R_i^{-1} \hat{g}_i \tag{43}$$

Here $\hat{\theta}^{(i)}$ is the parameter estimate after iteration number $i$. The search scheme is thus made up of the three entities

- $\mu_i$ step size
- $\hat{g}_i$ an estimate of the gradient $V_N'(\hat{\theta}^{(i)})$
- $R_i$ a matrix that modifies the search direction

It is useful to distinguish between two different minimization situations

**(i)** *Off-line* or *batch*: The update $\mu_i R_i^{-1} g_i^1$ is based on the whole available data record $Z^N$.

**(ii)** *On-line* or *recursive*: The update is based only on data up to sample $i$ ($Z^i$), (typically done so that the gradient estimate $\hat{g}_i$ is based only on data just before sample $i$.

We shall discuss these two modes separately below. First some general aspects will be treated.

**Search directions**

The basis for the local search is the gradient

$$V_N'(\theta) = \frac{dV_N(\theta)}{d\theta} = -\frac{1}{N}\sum_{t=1}^{N}(y(t) - \hat{y}(t|\theta))\psi(t,\theta) \tag{44}$$

where

$$\psi(t,\theta) = \frac{\partial}{\partial\theta}\hat{y}(t|\theta) \tag{45}$$

The gradient $\psi$ is in the general case a matrix with *dim* $\theta$ rows and *dim* $y$ columns. It is well known that gradient search for the minimum is inefficient, especially close to the minimum. Then it is optimal to use the *Newton search direction*

$$R^{-1}(\theta)V_N'(\theta) \tag{46}$$

where

$$\begin{aligned} R(\theta) = V_N''(\theta) = \frac{d^2V_N(\theta)}{d\theta^2} &= \frac{1}{N}\sum_{t=1}^{N}\psi(t,\theta)\psi^T(t,\theta) \\ &+ \frac{1}{N}\sum_{t=1}^{N}(y(t) - \hat{y}(t|\theta))\frac{\partial^2}{\partial\theta^2}\hat{y}(t|\theta) \end{aligned} \tag{47}$$

The true Newton direction will thus require that the second derivative

$$\frac{\partial^2}{\partial\theta^2}\hat{y}(t|\theta)$$

be computed. Also, far from the minimum, $R(\theta)$ need not be positive semidefinite. Therefore alternative search directions are more common in practice:

- *Gradient direction.* Simply take

$$R_i = I \tag{48}$$

- *Gauss-Newton direction.* Use

$$R_i = H_i = \frac{1}{N}\sum_{t=1}^{N} \psi(t, \hat{\theta}^{(i)})\psi^T(t, \hat{\theta}^{(i)}) \tag{49}$$

- *Levenberg-Marquard direction.* Use

$$R_i = H_i + \delta I \tag{50}$$

  where $H_i$ is defined by (49).

- *Conjugate gradient direction.* Construct the Newton direction from a sequence of gradient estimates. Loosely, think of $V_N''$ as constructed by difference approximation of $d$ gradients. The direction (46) is however constructed directly, without explicitly forming and inverting $V''$.

It is generally considered, [Dennis and Schnabel, 1983], that the Gauss-Newton search direction is to be preferred. For ill-conditioned problems the Levenberg-Marquard modification is recommended.

## On-line algorithms

The expressions (44) and (47) for the Gauss-Newton search clearly assume that the whole data set $Z^N$ is available during the iterations. If the application is of an off-line character, i.e., the model $\hat{g}_N$ is not required during the data acquisition, this is also the most natural approach.

However, many adaptive situations require on-line (or recursive) algorithms, where the data are processed as they are measured. (Such algorithms are in Neural Network contexts often also used in off-line situations.) Then the measured data record is concatenated with itself several times to create a (very) long record that is fed into the on-line algorithm. We may refer to [Ljung and Söderström, 1983] as a general reference for recursive parameter estimation algorithms. In [Solbrand et al., 1985] the use of such algorithms in the off-line case is discussed.

It is natural to consider the following algorithm as the basic one:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \mu_t R_t^{-1} \psi(t, \hat{\theta}(t-1)) \varepsilon(t, \hat{\theta}(t-1)) \tag{51}$$

$$\varepsilon(t, \theta) = y(t) - \hat{y}(t|\theta) \tag{52}$$

$$R_t = R_{t-1} + \mu_t [\psi(t, \hat{\theta}(t-1)) \psi^T(t, \hat{\theta}(t-1)) - R_{t-1}] \tag{53}$$

The reason is that if $\hat{y}(t|\theta)$ is linear in $\theta$, then (51) – (53), with $\mu_t = 1/t$, provides the analytical solution to the minimization problem (42). This also means that this is a natural algorithm close to the minimum, where a second order expansion of the criterion is a good approximation. In fact, it is shown in [Ljung and Söderström, 1983], that (51) – (53) in general gives an estimate $\hat{\theta}(t)$ with the same ("optimal") statistical, asymptotic properties as the true minimum to (42).

It should be mentioned that the quantities $\hat{y}(t|\hat{\theta}(t-1))$ and $\psi(t, \hat{\theta}(t-1))$ would normally (except in the linear regression case) require the whole data record to be computed. This would violate the recursiveness of the algorithm. In practical implementations these quantities are therefore replaced by recursively computed approximations. The idea behind these approximations is to use the defining equation for $\hat{y}(t|\theta)$ and $\psi(t, \theta)$ (which typically are recursive equations), and replace any appearance of $\theta$ with its latest available estimate. See [Ljung and Söderström, 1983] for more details.

Some averaged variants of (51) – (53) have also been discussed:

$$\hat{\hat{\theta}}(t) = \hat{\hat{\theta}}(t-1) + \mu_t R_t^{-1} \psi(t, \hat{\theta}(t-1)) \varepsilon(t, \hat{\theta}(t-1)) \tag{54}$$

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \rho_t [\hat{\hat{\theta}}(t) - \hat{\theta}(t-1)] \tag{55}$$

The basic algorithm (51) – (53) then corresponds to $\rho_t = 1$. Using $\rho_t < 1$ gives a so called "accelerated convergence" algorithm. It was introduced by [Polyak and Juditsky, 1992] and has then been extensively discussed by [Kushner and Yang, 1993] and others. The remarkable thing with this averaging is that we achieve the same asymptotic statistical properties of $\hat{\theta}(t)$ by (54) – (55) with $R_t = I$ (gradient search) as by (51) – (53) if

$$\rho_t = 1/t$$

$$\mu_t >> \rho_t \qquad \mu_t \to 0$$

It is thus an interesting alternative to (51) – (53), in particular if $\dim \theta$ is large so $R_t$ is a big matrix.

### Local Minima

A fundamental problem with minimization tasks like (42) is that $V_N(\theta)$ may have several or many local (non-global) minima, where local search algorithms may get caught. There is no easy solution to this problem. It is usually well worth the effort to find a good initial value $\theta^{(0)}$ where to start the iterations. Other than that, only various global search strategies are left, such as random search, random restarts, simulated annealing, and the genetic algorithm.

# 3 Linear Black Box Systems

## 3.1 Linear System Descriptions in General

### A linear System with Additive Disturbances

A linear system with additive disturbances $v(t)$ can be described by

$$y(t) = G(q)u(t) + v(t) \tag{56}$$

Here $u(t)$ is the input signal, and $G(q)$ is the transfer function from input to output $y(t)$. The symbol $q$ is the shift operator, so (56) should be interpreted as

$$y(t) = \sum_{k=0}^{\infty} g_k u(t-k) + v(t) = (\sum_{k=0}^{\infty} g_k q^{-k}) u(t) + v(t) \tag{57}$$

The disturbance $v(t)$ can in general terms be characterized by its *spectrum*, which is a description of its frequency content. It is often more convenient to describe $v(t)$ as being (thought of as) obtained by filtering a white noise source $e(t)$ through a linear filter $H(q)$:

$$v(t) = H(q)e(t) \tag{58}$$

This is, from a linear identification perspective, equivalent to describing $v(t)$ as a signal with spectrum

$$\Phi_v(\omega) = \lambda |H(e^{i\omega})|^2 \tag{59}$$

where $\lambda$ is the variance of the noise source $e(t)$. We shall assume that $H(q)$ is normalized to be monic, i.e.,

$$H(q) = 1 + \sum_{k=1}^{\infty} h_k q^{-k} \tag{60}$$

Putting all of this together, we arrive at the standard linear system description

$$y(t) = G(q)u(t) + H(q)e(t) \tag{61}$$

## Parameterized Linear Models

Now, if the transfer functions $G$ and $H$ in (61) are not known, we would introduce parameters $\theta$ in their description that reflect our lack of knowledge.

The exact way of doing this is the topic of the present section as well as of Section 5.

In any case the resulting, parameterized model will be described as

$$y(t) = G(q, \theta)u(t) + H(q, \theta)e(t) \tag{62}$$

The parameters $\theta$ can then be estimated from data using the general procedures described in Chapter 2.

## Predictors for Linear Models

Given a system description (62) and input-output data up to time $t - 1$,

$$y(s), u(s) \quad s \leq t - 1 \tag{63}$$

how shall we predict the next output value $y(t)$?

In the general case of (62) the prediction can be deduced in the following way: Divide (62) by $H(q, \theta)$:

$$H^{-1}(q, \theta)y(t) = H^{-1}(q, \theta)G(q, \theta)u(t) + e(t)$$

or

$$y(t) = [1 - H^{-1}(q, \theta)]y(t) + H^{-1}(q, \theta)G(q, \theta)u(t) + e(t) \tag{64}$$

In view of the normalization (60) we find that

$$1 - H^{-1}(q, \theta) = \frac{H(q, \theta) - 1}{H(q, \theta)} = \frac{1}{H(q, \theta)} \sum_{k=1}^{\infty} h_k q^{-k}$$

The expression $[1 - H^{-1}(q, \theta)]y(t)$ thus only contains old values of $y(s),\ s \leq t - 1$. The right side of (64) is thus known at time $t - 1$, with the exception of $e(t)$. The prediction of $y(t)$ is simply obtained from (64) by deleting $e(t)$:

$$\hat{y}(t|\theta) = [1 - H^{-1}(q, \theta)]y(t) + H^{-1}(q, \theta)G(q, \theta)u(t) \tag{65}$$

This is a general expression for how ready-made models predict the next value of the output, given old values of $y$ and $u$.

## A Characterization of the Limiting Model in a General Class of Linear Models

Let us apply the general limit result (27)-(28) to the linear model structure (62) (or (65)). If we choose a quadratic criterion $\ell(\varepsilon) = \varepsilon^2$ (in the scalar output case) then this result tells us, in the time domain, that the limiting parameter estimate is the one that minimizes the filtered prediction error variance (for the input used during the experiment.) Suppose that the data actually have been generated by

$$y(t) = G_0(q)u(t) + v(t) \tag{66}$$

Let $\Phi_u(\omega)$ be the input spectrum and $\Phi_v(\omega)$ be the spectrum for the additive disturbance $v$. Then the filtered prediction error can be written

$$\begin{aligned}\varepsilon_F(t,\theta) = \frac{L(q)}{H(q,\theta)}[y(t) - G(q,\theta)u(t)] = \\ \frac{L(q)}{H(q,\theta)}[(G_0(q) - G(q,\theta))u(t) + v(t)]\end{aligned} \tag{67}$$

By Parseval's relation, the prediction error variance can also be written as an integral over the spectrum of the prediction error. This spectrum, in turn, is directly obtained from (67), so the limit estimate $\theta^*$ in (28) can also be defined as

$$\begin{aligned}\theta^* = \arg\min_\theta \Bigg[\int_{-\pi}^{\pi} |G_0(e^{i\omega}) - G(e^{i\omega},\theta)|^2 \frac{\Phi_u(\omega)|L(e^{i\omega})|^2}{|H(e^{i\omega},\theta)|^2} d\omega \\ + \int_{-\pi}^{\pi} \Phi_v(\omega)|L(e^{i\omega})|^2/|H(e^{i\omega},\theta)|^2 d\omega\Bigg]\end{aligned} \tag{68}$$

If the noise model $H(q,\theta) = H_*(q)$ does not depend on $\theta$ (as in the output error model (75)) the expression (68) thus shows that the resulting model

$G(e^{i\omega}, \theta^*)$ will give that frequency function in the model set that is closest to the true one, in a quadratic frequency norm with weighting function

$$Q(\omega) = \Phi_u(\omega)|L(e^{i\omega})|^2/|H_*(e^{i\omega})|^2 \tag{69}$$

This shows clearly that the fit can be affected by the choice of prefilter $L$, the input spectrum $\Phi_u$ and the noise model $H_*$.

## 3.2 Linear, Ready-made Models

Sometimes we are faced with systems or subsystems that cannot be modeled based on physical insights. The reason may be that the function of the system or its construction is unknown or that it would be too complicated to sort out the physical relationships. It is then possible to use standard models, which by experience are known to be able to handle a wide range of different system dynamics. Linear systems constitute the most common class of such standard models. From a modeling point of view these models thus serve as *ready-made models*: tell us the size (model order), and it should be possible to find something that fits (to data).

### A Family of Transfer Function Models

A very natural approach is to describe $G$ and $H$ in (62) as rational transfer functions in the shift (delay) operator with unknown numerator and denominator polynomials.

We would then have

$$G(q, \theta) = \frac{B(q)}{F(q)} = \frac{b_1 q^{-nk} + b_2 q^{-nk-1} + \cdots + b_{nb} q^{-nk-nb+1}}{1 + f_1 q^{-1} + \cdots + f_{nf} q^{-nf}}, \tag{70}$$

Then

$$\eta(t) = G(q, \theta) u(t) \tag{71}$$

is a shorthand notation for the relationship

$$\begin{aligned} &\eta(t) + f_1\eta(t-1) + \cdots + f_{nf}\eta(t-nf) \\ &\quad = b_1u(t-nk) + \cdots + b_{nb}(t-(nb+nk-1)) \end{aligned} \tag{72}$$

There is also a time delay of $nk$ samples. We assume assume, for simplicity, that the sampling interval $T$ is one time unit.

In the same way the disturbance transfer function can be written

$$H(q,\theta) = \frac{C(q)}{D(q)} = \frac{1 + c_1q^{-1} + \cdots + c_{nc}q^{-nc}}{1 + d_1q^{-1} + \cdots + d_{nd}q^{-nd}} \tag{73}$$

The parameter vector $\theta$ thus contains the coefficients $b_i$, $c_i$, $d_i$, and $f_i$ of the transfer functions. This ready-made model is thus described by five structural parameters: $nb$, $nc$, $nd$, $nf$, and $nk$. When these have been chosen, it remains to adjust the parameters $b_i$, $c_i$, $d_i$, and $f_i$ to data. This is done with the methods of Section 2. The ready-made model (70)-(73) gives

$$y(t) = \frac{B(q)}{F(q)}u(t) + \frac{C(q)}{D(q)}e(t) \tag{74}$$

and is known as the *Box-Jenkins (BJ) model*, after the statisticians G. E. P. Box and G. M. Jenkins.

An important special case is when the properties of the disturbance signals are not modeled, and the noise model $H(q)$ is chosen to be $H(q) \equiv 1$; that is, $nc = nd = 0$. This special case is known as an *output error (OE) model* since the noise source $e(t)$ will then be the difference (error) between the actual output and the noise-free output:

$$y(t) = \frac{B(q)}{F(q)}u(t) + e(t) \tag{75}$$

A common variant is to use the same denominator for $G$ and $H$:

$$F(q) = D(q) = A(q) = 1 + a_1q^{-1} + \cdots + a_{na}a^{-na} \tag{76}$$

Multiplying both sides of (74) by $A(q)$ then gives

$$A(q)y(t) = B(q)u(t) + C(q)e(t) \tag{77}$$

This ready-made model is known as the *ARMAX model.* The name is derived from the fact that $A(q)y(t)$ represents an AutoRegression and $C(q)e(t)$ a Moving Average of white noise, while $B(q)u(t)$ represents an eXtra input (or with econometric terminology, an eXogenous variable).

Physically, the difference between ARMAX and BJ models is that the noise and input are subjected to the same dynamics (same poles) in the ARMAX case. This is reasonable if the dominating disturbances enter early in the process (together with the input). Consider for example an airplane where the disturbances from wind gusts give rise to the same type of forces on the airplane as the deflections of the control surfaces.

Finally, we have the special case of (77) that $C(q) \equiv 1$, that is, $nc = 0$

$$A(q)y(t) = B(q)u(t) + e(t) \tag{78}$$

which with the same terminology is called an *ARX model*, and which we discussed at length in Section 1.3.

Figure 1 shows the most common model structures.

To use these ready-made models, decide on the orders $na, nb, nc, nd$, $nf$, and $nk$ and let the computer pick the best model in the class thus defined. The obtained model is then scrutinized, and it might be found that other order must also be tested.

A relevant question is how to use the freedom that the different model structures give. Each of the BJ, OE, ARMAX, and ARX structures offer their own advantages, and we will discuss them in Section 7.2.

## Prediction

Starting with model (74), it is possible to predict what the output $y(t)$ will be, based on measurements of $u(s),\ y(s)\ s \leq t-1$, using the general formula

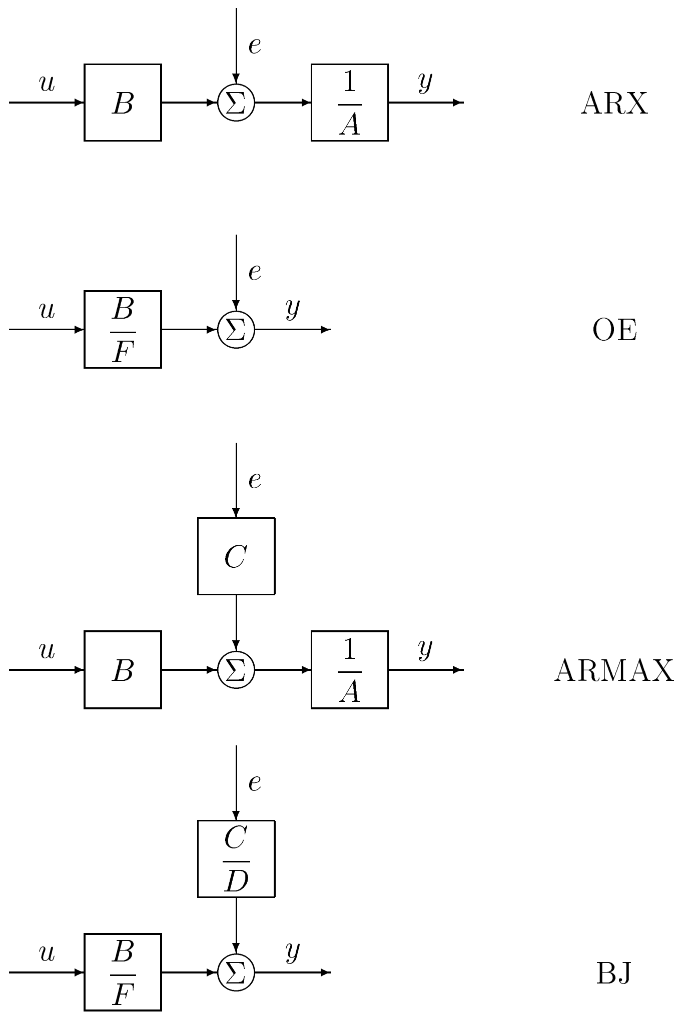

Figure 1: Model structures.

(65) It is easiest to calculate the prediction for the OE-case, $H(q, \theta) \equiv 1$, when we obtain the model

$$y(t) = G(q, \theta)u(t) + e(t)$$

with the natural prediction $(1 - H^{-1} = 0)$

$$\hat{y}(t|\theta) = G(q, \theta)u(t) \tag{79}$$

From the ARX case (78) we obtain

$$\begin{aligned} y(t) = &-a_1 y(t-1) - \cdots - a_{na} y(t-na) \\ &+ b_1 u(t-nk) + \cdots + b_{nb} u(t-nk-nb+1) + e(t) \end{aligned} \tag{80}$$

and the prediction (delete $e(t)$!)

$$\begin{aligned} \hat{y}(t|\theta) = &-a_1 y(t-1) - \cdots - a_{na} y(t-na) \\ &+ b_1 u(t-nk) + \cdots + b_{nb} u(t-nk-nb+1) \end{aligned} \tag{81}$$

Note the difference between (79) and (81). In the OE model the prediction is based entirely on the input $\{u(t)\}$, whereas the ARX model also uses old values of the output.

## Linear Regression

Both tailor-made and ready-made models describe how the predicted value of $y(t)$ depends on old values of $y$ and $u$ and on the parameters $\theta$. We denote this prediction by

$$\hat{y}(t|\theta)$$

See (65). In general this can be a rather complicated function of $\theta$. The estimation work is considerably easier if the prediction is a linear function of $\theta$:

$$\hat{y}(t|\theta) = \theta^T \varphi(t) \tag{82}$$

Here $\theta$ is a column vector that contains the unknown parameters, while $\varphi(t)$ is a column vector formed by old inputs and outputs. Such a model structure is called a *linear regression*. We discussed such models in Section 1.3, and noted that the ARX model (78) is one common model of the linear regression type. Linear regression models can also be obtained in several other ways. See Example 2.

# 4 Special Estimation Techniques for Linear Black Box Models

An important feature of a linear, time invariant system is that it is entirely characterized by its *impulse response*. So if we know the system's response to an impulse, we will also know its response to any input. Equivalently, we could study the *frequency response*, which is the Fourier transform of the impulse response.

In this section we shall consider estimation methods for linear systems, that do not use particular model parameterizations. First, in Section 4.1, we shall consider direct methods to determine the impulse response and the frequency response, by simply applying the definitions of these concepts.

In section 4.2 methods for estimating the impulse response by correlation analysis will be described, and in Section 4.3 spectral analysis for frequency function estimation will be discussed. Finally, in Section 4.4 a recent method to estimate general linear systems (of given order, by unspecified structure) will be described.

## 4.1 Transient and Frequency Analysis

### Transient Analysis

The first step in modeling is to decide which quantities and variables are important to describe what happens in the system. A simple and common kind of experiment that shows how and in what time span various variables affect each other is called *step-response analysis* or *transient analysis.* In such experiments the inputs are varied (typically one at a time) as a step: $u(t) = u_0$, $t < t_0$; $u(t) = u_1$, $t \geq t_0$. The other measurable variables in the system are recorded during this time. We thus study the *step response* of the system. An alternative would be to study the impulse response of the system by letting the input be a pulse of short duration. From such measurements, information of the following nature can be found:

1. The variables affected by the input in question. This makes it easier to draw block diagrams for the system and to decide which influences can be neglected.
2. The time constants of the system. This also allows us to decide which relationships in the model can be described as static (that is, they have significantly faster time constants than the time scale we are working with.
3. The characteristic (oscillatory, poorly damped, monotone, and the like) of the step responses, as well as the levels of static gains. Such information is useful when studying the behavior of the final model in simulation. Good agreement with the measured step responses should give a certain confidence in the model.

**Frequency Analysis**

If a linear system has the transfer function $G(q)$ and the input is

$$u(t) = u_0 \cos \omega k T, \quad (k-1)T \leq t \leq kT \tag{83}$$

then the output after possible transients have faded away will be

$$y(t) = y_0 \cos(\omega t + \varphi), \quad \text{for} \quad t = T, 2T, 3T, \ldots \tag{84}$$

where

$$y_0 = |G(e^{i\omega T})| \cdot u_0 \tag{85}$$

$$\varphi = \arg G(e^{i\omega T}) \tag{86}$$

If the system is driven by the input (83) for a certain $u_0$ and $\omega_1$ and we measure $y_0$ and $\varphi$ from the output signal, it is possible to determine the complex number $G(e^{i\omega_1 T})$ using (85)–(86). By repeating this procedure for a number of different $\omega$, we can get a good estimate of the frequency function $G(e^{i\omega T})$. This method is called *frequency analysis.* Sometimes it is possible to see or measure $u_0$, $y_0$, and $\varphi$ directly from graphs of the input and output signals. Most of the time, however, there will be noise and irregularities that make it difficult to determine $\varphi$ directly. A suitable procedure is then to correlate the output with $\cos \omega t$ and $\sin \omega t$.

## 4.2 Estimating Impulse Responses by Correlation Analysis

It is not necessary to use an impulse as input to estimate the impulse response of a system directly. That can also be done by correlation techniques. To explain how these work, let us first define correlation functions.

The *cross covariance function* between two signals $y$ and $u$ is defined as the covariance between the random variables $y(t)$ and $u(t-\tau)$, viewed as a function of the time difference $\tau$:

$$R_{yu}(\tau) = E(y(t) - Ey(t))(u(t-\tau) - Eu(t-\tau)) \tag{87}$$

It is implicitly assumed here that the indicated expectation does not depend on absolute time $t$. This is the same as saying that the signals are (weakly) *stationary.*

Just as in the case (32), expectation can be replaced by sample means:

$$m_y = \lim_{N\to\infty} \frac{1}{N} \sum_{t=1}^{N} y(t) \tag{88}$$

$$R_{yu}(\tau) = \lim_{N\to\infty} \frac{1}{N} \sum_{t=1}^{N} (y(t) - m_y)(u(t-\tau) - m_u) \tag{89}$$

As soon as we use the term covariance function, there is always an implied assumption that the involved signals are such that either (87) or (89) is well defined.

The cross correlation signal between a signal $u$ and itself, i.e. $R_{uu}(\tau) = R_u(\tau)$ is called the (auto) *covariance function* of the signal.

We shall say that two signals are *uncorrelated* if their cross covariance function is identically equal to zero.

Let us consider the general linear model (56), and assume that the input $u$ and the noise $v$ are uncorrelated:

$$y(t) = \sum_{k=0}^{\infty} g_k u(t-k) + v(t) \tag{90}$$

The cross covariance function between $u$ and $y$ is then

$$\begin{aligned} R_{yu}(\tau) = Ey(t)u(t-\tau) &= \sum_{k=0}^{\infty} g_k Eu(t-k)u(t-\tau) \\ + Ev(t)u(t-\tau) &= \sum_{k=0}^{\infty} g_k R_u(\tau-k) \end{aligned} \tag{91}$$

If the input is white noise,

$$R_u(\tau) = \begin{cases} \lambda, \tau = 0 \\ 0, \tau \neq 0 \end{cases}$$

we obtain

$$R_{yu}(\tau) = \lambda g_\tau \tag{92}$$

*The cross covariance function $R_{yu}(\tau)$ will thus be proportional to the impulse response.* Of course, this function is not known, but it can be estimated in an obvious way from observed inputs and outputs as the corresponding sample mean:

$$\hat{R}_{yu}^N(\tau) = \frac{1}{N} \sum_{t=1}^{N} y(t)u(t-\tau) \tag{93}$$

In this way we also obtain an estimate of the impulse response:

$$\hat{g}_\tau^N = \frac{1}{\lambda}\hat{R}_{yu}^N(\tau) \tag{94}$$

If we cannot choose the input ourselves, and it is non white, it would be possible to estimate its covariance function as $\hat{R}_u^N(\tau)$, analogous to (93), and then solve for $g_k$ from (91) where $R_u$ and $R_{uy}$ have been replaced by the corresponding estimates. However, a better and more common way is the following: First note that if both input and output are filtered through the same filter

$$y_F(t) = L(q)y(t) \qquad\qquad u_F(t) = L(q)u(t) \tag{95}$$

then the filtered signals will be related by the same impulse response as in (90):

$$y_F(t) = \sum_{k=1}^{\infty} g_k u_F(t-k) + v_F(t) \tag{96}$$

Now, for any given input $u(t)$ the the process, we can choose the filter $L$ so that the signal $\{u_F(t)\}$ will be as white as possible. Such a filter is called a *whitening filter*. It is often computed by describing $u(t)$ as an AR process (This is an ARX model without an input, cf Section 1.3): $A(q)u(t) = e(t)$. The polynomial $A(q) = L(q)$ can then be estimated using the least squares method. (See Section 1.3). We can now use the estimate (94) applied to the filtered signals.

## 4.3 Estimating the Frequency Response by Spectral Analysis

### Definitions

The *cross spectrum* between two (stationary) signals $u(t)$ and $y(t)$ is defined as the Fourier transform of their cross covariance function, provided this exists:

$$\Phi_{yu}(\omega) = \sum_{\tau=-\infty}^{\infty} R_{yu}(\tau) e^{-i\omega\tau} \tag{97}$$

where $R_{yu}(\tau)$ is defined by (87) or (89). The (auto) *spectrum* $\Phi_u(\omega)$ of a signal $u$ is defined as $\Phi_{uu}(\omega)$, i.e. as its cross spectrum with itself.

The spectrum describes the frequency contents of the signal. The connection to more explicit Fourier techniques is evident by the following relationship

$$\Phi_u(\omega) = \lim_{N\to\infty} \frac{1}{N} |U_N(\omega)|^2 \tag{98}$$

where $U_N$ is the discrete time Fourier transform

$$U_N(\omega) = \sum_{t=1}^{N} u(t) e^{i\omega t} \tag{99}$$

The relationship (98) is shown, e.g. in [Ljung and Glad, 1994].

Consider now the general linear model (56):

$$y(t) = G(q)u(t) + v(t) \tag{100}$$

It is straightforward to show that the relationships between the spectra and cross spectra of $y$ and $u$ (provided $u$ and $v$ are uncorrelated) is given by

$$\Phi_{yu}(\omega) = G(e^{i\omega})\Phi_u(\omega) \tag{101}$$

$$\Phi_y(\omega) = |G(e^{i\omega})|^2 \Phi_u(\omega) + \Phi_v(\omega) \tag{102}$$

It is easy to see how the transfer function $G(e^{i\omega})$ and the noise spectrum $\phi_v(\omega)$ can be estimated using these expressions, if only we have a method to estimate cross spectra.

## Estimation of Spectra

The spectrum is defined as the Fourier transform of the correlation function. A natural idea would then be to take the transform of the estimate $\hat{R}^N_{yu}(\tau)$ in (93). That will not work in most cases, though. The reason could be described as follows: The estimate $\hat{R}^N_{yu}(\tau)$ is not reliable for large $\tau$, since it is based on only a few observations. These "bad" estimates are mixed with good ones in the Fourier transform, thus creating an overall bad estimate. It is better to introduce a weighting, so that correlation estimates for large lags $\tau$ carry a smaller weight:

$$\hat{\Phi}^N_{yu}(\omega) = \sum_{\ell=-\gamma}^{\gamma} \hat{R}^N_{yu}(\ell) \cdot w_\gamma(\ell) e^{-i\ell\omega} \tag{103}$$

This spectral estimation method is known as the The *Blackman-Tukey approach.* Here $w_\gamma(\ell)$ is a window function that decreases with $|\tau|$. This function controls the trade-off between *frequency resolution* and *variance of the estimate.* A function that gives significant weights to the correlation at large lags will be able to provide finer frequency details (a longer time span is covered). At the same time it will have to use "bad" estimates, so the statistical quality (the variance) is poorer. We shall return to this trade-off in a moment. How should we choose the shape of the window function $w_\gamma(\ell)$? There is no optimal solution to this problem, but the most common window used in spectral analysis is the *Hamming window*:

$$\begin{array}{ll} w_\gamma(k) = \frac{1}{2}(1 + \cos \frac{\pi k}{\gamma}) & |k| < \gamma \\ w_\gamma(k) = 0 & |k| \geq \gamma \end{array} \tag{104}$$

From the spectral estimates $\Phi_u$, $\Phi_y$ and $\Phi_{yu}$ obtained in this way, we can now use (101) to obtain a natural estimate of the frequency function $G(e^{i\omega})$:

$$\hat{G}_N(e^{i\omega}) = \frac{\hat{\Phi}^N_{yu}(\omega)}{\hat{\Phi}^N_u(\omega)} \tag{105}$$

Furthermore, the disturbance spectrum can be estimated from (102) as

$$\hat{\Phi}_v^N(\omega) = \hat{\Phi}_y^N(\omega) - \frac{|\hat{\Phi}_{yu}^N(\omega)|^2}{\hat{\Phi}_u^N(\omega)} \tag{106}$$

To compute these estimates, the following steps are performed:

**Algorithm SPA** (107)

1. Collect data $y(k)$, $u(k)$ $k = 1, \ldots, N$.
2. Subtract the corresponding sample means form the data. This will avoid bad estimates at very low frequencies.
3. Choose the width of the lag window $w_\gamma(k)$.
4. Compute $\hat{R}_y^N(k)$, $\hat{R}_u^N(k)$, and $\hat{R}_{yu}^N(k)$ for $|k| \leq \gamma$ according to (93).
5. Form the spectral estimates $\hat{\Phi}_y^N(\omega)$, $\hat{\Phi}_u^N(\omega)$, and $\hat{\Phi}_{yu}^N(\omega)$ according to (103) and analogous expressions.
6. Form (105) and possibly also (106).

The user only has to choose $\gamma$. A good value for systems without sharp resonances is $\gamma = 20$ to 30. Larger values of $\gamma$ may be required for systems with narrow resonances.

## Quality of the Estimates

The estimates $\hat{G}_N$ and $\hat{\Phi}_w^N$ are formed entirely from estimates of spectra and cross spectra. Their properties will therefore be inherited from the properties of the spectral estimates. For the Hamming window with width $\gamma$, it can be shown that the frequency resolution will be about

$$\frac{\pi}{\gamma\sqrt{2}} \qquad \text{radians/time unit} \tag{108}$$

This means that details in the true frequency function that are finer than this expression will be smeared out in the estimate. It is also possible to show that the estimate's variances satisfy

$$\text{Var } \hat{G}_N(i\omega) \approx 0.7 \cdot \frac{\gamma}{N} \cdot \frac{\Phi_v(\omega)}{\Phi_u(\omega)} \tag{109}$$

and

$$\text{Var } \hat{\Phi}_v^N(\omega) \approx 0.7 \cdot \frac{\gamma}{N} \cdot \Phi_v^2(\omega) \tag{110}$$

[Variance" here refers to taking expectation over the noise sequence $v(t)$.] Note that the relative variance in (109) typically increases dramatically as $\omega$ tends to the Nyquist frequency. The reason is that $|G(i\omega)|$ typically decays rapidly, while the noise-to-signal ratio $\Phi_v(\omega)/\Phi_u(\omega)$ has a tendency to increase as $\omega$ increases. In a Bode diagram the estimates will thus show considerable fluctuations at high frequencies. Moreover, the constant frequency resolution (108) will look thinner and thinner at higher frequencies in a Bode diagram due to the logarithmic frequency scale.

See [Ljung and Glad, 1994] for a more detailed discussion.

### Choice of Window Size

The choice of $\gamma$ is a pure trade-off between frequency resolution and variance (variability). For a spectrum with narrow resonance peaks it is thus necessary to choose a large value of $\gamma$ and accept a higher variance. For a more flat spectrum, smaller values of $\gamma$ will do well. In practice a number of different values of $\gamma$ are tried out. Often we start with a small value of $\gamma$ and increase it successively until an estimate is found that balances the trade-off between frequency resolution (true details) and variance (random fluctuations). A typical value for spectra without narrow resonances is $\gamma$= 20–30.

## 4.4 Subspace Estimation Techniques for State Space Models

A linear system can always be represented in state space form:

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) + w(t) \\ y(t) &= Cx(t) + Du(t) + e(t) \end{aligned} \tag{111}$$

We assume that we have no insight into the particular structure, and we would just estimate any matrices $A, B, C$, and $D$, that give a good description of the input-output behavior of the system. This is not without problems, among other things because there are an infinite number of such matrices that describe the same system (the similarity transforms). The coordinate basis of the state-space realization thus needs to be fixed.

Let us for a moment assume that not only are $u$ and $y$ measured, but also the sequence of state vectors $x$. This would, by the way, fix the state-space realization coordinate basis. Now, with known $u, y$ and $x$, the model (111) becomes a linear regression: the unknown parameters, all of the matrix entries in all the matrices, mix with measured signals in linear combinations. To see this clearly, let

$$\begin{aligned} Y(t) &= \begin{pmatrix} x(t+1) \\ y(t) \end{pmatrix} \\ \Theta &= \begin{pmatrix} A & B \\ C & D \end{pmatrix} \\ \Phi(t) &= \begin{pmatrix} x(t) \\ u(t) \end{pmatrix} \\ E(t) &= \begin{pmatrix} w(t) \\ e(t) \end{pmatrix} \end{aligned}$$

Then, (111) can be rewritten as

$$Y(t) = \Theta\Phi(t) + E(t) \tag{112}$$

From this all the matrix elements in $\Theta$ can be estimated by the simple least squares method, as described in Section 1.3. The covariance matrix for $E(t)$

can also be estimated easily as the sample sum of the model residuals. That will give the covariance matrices for $w$ and $e$, as well as the cross covariance matrix between $w$ and $e$. These matrices will, among other things, allow us to compute the Kalman filter for (111). Note that all of the above holds without changes for multivariable systems, i.e., when the output and input signals are vectors.

The only remaining problem is where to get the state vector sequence $x$ from. It has long been known, e.g., [Rissanen, 1974], [Akaike, 1974b], that all state vectors $x(t)$ that can be reconstructed from input-output data in fact are linear combinations of the components of the $n$ $k$-step ahead output predictors

$$\hat{y}(t+k|t), \quad k = \{1, 2, \ldots, n\} \tag{113}$$

where $n$ is the model order (the dimension of $x$). See also Appendix 4.A in [Ljung, 1987]. We could then form these predictors, and select a basis among their components:

$$x(t) = L \begin{pmatrix} \hat{y}(t+1|t) \\ \vdots \\ \hat{y}(t+n|t) \end{pmatrix} \tag{114}$$

The choice of $L$ will determine the basis for the state-space realization, and is done in such a way that it is well conditioned. The predictor $\hat{y}(t+k|t)$ is a linear function of $u(s), y(s), \quad 1 \leq s \leq t$ and can efficiently be determined by linear projections directly on the input output data. (There is one complication in that $u(t+1), \ldots, u(t+k)$ should not be predicted, even if they affect $y(t+k)$.)

What we have described now is the *subspace projection* approach to estimating the matrices of the state-space model (111), including the basis for the representation and the noise covariance matrices. There are a number of variants of this approach. See among several references, e.g. [Overschee and DeMoor, 1994], [Larimore, 1983]

The approach gives very useful algorithms for model estimation, and is particularly well suited for multivariable systems. The algorithms also allow numerically very reliable implementations. At present, the asymptotic properties of the methods are not fully investigated, and the general results quoted in Section 2.2 are not directly applicable. Experience has shown, however, that confidence intervals computed according to the general asymptotic theory, are good approximations. One may also use the estimates obtained by a subspace method as initial conditions for minimizing the prediction error criterion (24).

## 5 Physically parameterized models

So far we have treated the parameters $\theta$ only as vehicles to give reasonable flexibility to the transfer functions in the general linear model (62). This model can also be arrived at from other considerations.

Consider a continuous time state space model

$$\dot{x}(t) = A(\theta)x(t) + B(\theta)u(t) \tag{115a}$$

$$y(t) = C(\theta)x(t) + v(t) \tag{115b}$$

Here $x(t)$ is the state vector and typically consists of physical variables (such as positions and velocities etc). The state space matrices $A$, $B$ and $C$ are parameterized by the parameter vector $\theta$, reflecting the physical insight we have into the process. The parameters could be physical constants (resistance, heat transfer coefficients, aerodynamical derivatives etc) whose values are not known. They could also reflect other types of insights into the system's properties.

**Example 8.4** *An electric motor*

*Consider an electric motor with the input $u$ being the applied voltage and the output $y$ being the angular position of the motor shaft.*

*A first, but reasonable approximation of the motor's dynamics is as a first* *order system from voltage to angular velocity, followed by an integrator:*

$$G(s) = \frac{b}{s(s+a)}$$

*If we select the state variables*

$$x(t) = \begin{pmatrix} y(t) \\ \dot{y}(t) \end{pmatrix}$$

*we obtain the state space form*

$$\begin{aligned} \dot{x} &= \begin{pmatrix} 0 & 1 \\ 0 & -a \end{pmatrix} x + \begin{pmatrix} 0 \\ b \end{pmatrix} u \\ y &= \begin{pmatrix} 1 & 0 \end{pmatrix} x + v \end{aligned} \tag{116}$$

*where $v$ denotes disturbances and noise. In this case we thus have*

$$\begin{aligned} \theta &= \begin{pmatrix} a \\ b \end{pmatrix} \\ A(\theta) &= \begin{pmatrix} 0 & 1 \\ 0 & -a \end{pmatrix} \qquad B(\theta) = \begin{pmatrix} 0 \\ b \end{pmatrix} \\ C &= \begin{pmatrix} 1 & 0 \end{pmatrix} \end{aligned} \tag{117}$$

*The parameterization reflects our insight that the system contains an integration, but is in this case not directly derived from detailed physical modeling. Basic physical laws would in this case have given us how $\theta$ depends on physical constants, such as resistance of the wiring, amount of inertia, friction coefficients and magnetic field constants.* □

Now, how do we fit a continuous-time model (115a) to sampled observed data? If the input $u(t)$ has been piecewise constant over the sampling interval

$$u(t) = u(kT) \qquad kT \le t < (k+1)T$$

then the states, inputs and outputs at the sampling instants will be represented by the discrete time model

$$\begin{aligned} x((k+1)T) &= \bar{A}(\theta)x(kT) + \bar{B}(\theta)u(kT) \\ y(kT) &= C(\theta)x(kT) + v(kT) \end{aligned} \tag{118}$$

where

$$\bar{A}(\theta) = e^{A(\theta)T}, \quad \bar{B}(\theta) = \int_0^T e^{A(\theta)\tau} B(\theta) d\tau \tag{119}$$

This follows from solving (115) over one sampling period. We could also further model the added noise term $v(kT)$ and represent the system in the innovations form

$$\begin{aligned} \bar{x}((k+1)T) &= \bar{A}(\theta)\bar{x}(kT) + \bar{B}(\theta)u(kT) + \bar{K}(\theta)e(kT) \\ y(kT) &= C(\theta)\bar{x}(kT) + e(kT) \end{aligned} \tag{120}$$

where $\{e(kT)\}$ is white noise. The step from (118) to (120) is really a standard Kalman filter step: $\bar{x}$ will be the one-step ahead predicted Kalman states. A pragmatic way to think about it is as follows: In (118) the term $v(kT)$ may not be white noise. If it is colored we may separate out that part of $v(kT)$ that cannot be predicted from past values. Denote this part by $e(kT)$: it will be the *innovation.* The other part of $v(kT)$ – the one that can be predicted – can then be described as a combination of earlier innovations, $e(\ell T)\ \ell < k$. Its effect on $y(kT)$ can then be described via the states, by changing them from $x$ to $\bar{x}$, where $\bar{x}$ contains additional states associated with getting $v(kT)$ from $e(\ell T),\ k \leq \ell$.

Now (120) can be written in input – output from as (let $T = 1$)

$$y(t) = G(q,\theta)u(t) + H(q,\theta)e(t) \tag{121}$$

with

$$\begin{aligned} G(q,\theta) &= C(\theta)(qI - \bar{A}(\theta))^{-1}\bar{B}(\theta) \\ H(q,\theta) &= I + C(\theta)(qI - \bar{A}(\theta))^{-1}\bar{K}(\theta) \end{aligned} \tag{122}$$

We are thus back at the basic linear model (62). The parameterization of $G$ and $H$ in terms of $\theta$ is however more complicated than the ones we discussed in Section 3.2.

The general estimation techniques, model properties (including the characterization (68)), algorithms, etc., apply exactly as described in Section 2.

From these examples it is also quite clear that non-linear models with unknown parameters can be approached in the same way. We would then typically arrive at a a structure

$$\begin{aligned}\dot{x}(t) &= f(x(t), u(t), \theta) \\ y(t) &= h(x(t), u(t), \theta) + v(t) \end{aligned} \tag{123}$$

In this model, all noise effects are collected as additive output disturbances $v(t)$ which is a restriction, but also a very helpful simplification. If we define $\hat{y}(t|\theta)$ as the simulated output response to (123), for a given input, ignoring the noise $v(t)$, everything that was said in Section 2 about parameter estimation, model properties, etc. is still applicable.

# 6 Non-linear Black Box Models

In this section we shall describe the basic ideas behind model structures that have the capability to cover any non-linear mapping from past data to the predicted value of $y(t)$. Recall that we defined a general model structure as a parameterized mapping in (19):

$$\hat{y}(t|\theta) = g(\theta, Z^{t-1}) \tag{124}$$

We shall consequently allow quite general non-linear mappings $g$. This section will deal with some general principles for how to construct such mappings, and will cover Artificial Neural Networks as a special case. See [Sjöberg et al., 1995] and [Juditsky et al., 1995] for recent and more comprehensive surveys.

## 6.1 Non-Linear Black-Box Structures

Now, the model structure family (124) is really too general, and it turns out to be useful to write $g$ as a concatenation of two mappings: one that takes the increasing number of past observations $Z^{t-1}$ and maps them into a finite dimensional vector $\varphi(t)$ of fixed dimension and one that takes this vector to the space of the outputs:

$$\hat{y}(t|\theta) = g(\theta, Z^{t-1}) = g(\varphi(t), \theta) \tag{125}$$

where

$$\varphi(t) = \varphi(Z^{t-1}) \tag{126}$$

Let the dimension of $\varphi$ be $d$. As before, we shall call this vector the *regression vector* and its components will be referred to as the *regressors.* We also allow the more general case that the formation of the regressors is itself parameterized:

$$\varphi(t) = \varphi(Z^{t-1}, \eta) \tag{127}$$

which we for short write $\varphi(t, \eta)$. For simplicity, the extra argument $\eta$ will however be used explicitly only when essential for the discussion.

The choice of the non-linear mapping in (124) has thus been reduced to two partial problems for dynamical systems:

1. How to choose the non-linear mapping $g(\varphi)$ from the regressor space to the output space (*i.e.*, from $R^d$ to $R^p$).

2. How to choose the regressors $\varphi(t)$ from past inputs and outputs.

The second problem is the same for all dynamical systems, and it turns out that the most useful choices of regression vectors are to let them contain past inputs and outputs, and possibly also past predicted/simulated outputs. The regression vector will thus be of the character (4). We now turn to the first problem.

## 6.2 Non-Linear Mappings: Possibilities

### Function Expansions and Basis Functions

The non-linear mapping

$$g(\varphi, \theta) \tag{128}$$

goes from $R^d$ to $R^p$ for any given $\theta$. At this point it does not matter how the regression vector $\varphi$ is constructed. It is just a vector that lives in $R^d$.

It is natural to think of the parameterized function family as function expansions:

$$g(\varphi, \theta) = \sum \theta(k) g_k(\varphi) \tag{129}$$

where $g_k$ are the *basis functions* and the coefficients $\theta(k)$ are the “coordinates” of $g$ in the chosen basis.

Now, the only remaining question is: How to choose the basis functions $g_k$? Depending on the support of $g_k$ (*i.e.* the area in $R^d$ for which $g_k(\varphi)$ is (practically) non-zero) we shall distinguish between three types of basis functions

- Global basis functions
- Semi-global or ridge-type basis functions
- Local basis functions

A typical and classical global basis function expansion would then be the Taylor series, or polynomial expansion, where $g_k$ would contain multinomials in the components of $\varphi$ of total degree $k$. Fourier series are also relevant examples. We shall however not discuss global basis functions here any further. Experience has indicated that they are inferior to the semi-local and local ones in typical practical applications.

## Local Basis Functions

Local basis functions have their support only in some neighborhood of a given point. Think (in the case of $p$=1) of the indicator function for the unit cube:

$$\kappa(\varphi) = 1 \text{ if } |\varphi_k| \leq 1 \ \forall k\,, \text{ and } 0 \text{ otherwise} \tag{130}$$

By scaling the cube and placing it at different locations we obtain the functions

$$g_k(\varphi) = \kappa(\alpha_k * (\varphi - \beta_k)) \tag{131}$$

By allowing $\alpha$ to be a vector of the same dimension as $\varphi$ and interpreting the multiplication $*$ as component-wise multiplication (like ".$*$" in MATLAB) we may also reshape the cube to be any parallelepiped. The parameters $\alpha$ are thus *scaling* or *dilation* parameters while $\beta$ determine *location* or *translation*. For notational convenience we write

$$g_k(\varphi) = \kappa(\alpha_k * (\varphi - \beta_k)) = \kappa(\rho_k \cdot \varphi) \tag{132}$$

where

$$\rho_k = [\alpha_k, \alpha_k * \beta_k]$$

In the last equality, with some abuse of notation, we expanded the regression vector $\varphi$ to contain some "1":s. This is to stress the point that the argument of the basic function $\kappa$ is bilinear in the scale and location parameters $\rho_k$ and in the regression vector $\varphi$. The notation $\rho_k \cdot \varphi$ indicates this.

This choice of $g_k$ in (129) gives functions that are piecewise constant over areas in $R^d$ that can be chosen arbitrarily small by proper choice of the scaling parameters. It should be fairly obvious that such functions $g_k$ can approximate any reasonable function arbitrarily well.

Now it is also reasonable that the same will be true for any other localized function, such as the Gaussian bell function:

$$\kappa(\varphi) = e^{-|\varphi|^2} \tag{133}$$

### Ridge-type Basis Functions

A useful alternative is to let the basis functions be local in one direction of the $\varphi$-space and global in the others. This is achieved quite analogously to (131) as follows. Let $\sigma(x)$ be a local function from $R$ to $R$. Then form

$$g_k(\varphi) = \sigma(\alpha_k^T(\varphi - \beta_k)) = \sigma(\alpha_k^T\varphi + \gamma_k) = \sigma(\rho_k \cdot \varphi) \tag{134}$$

where the scalar $\gamma_k = -\alpha_k^T\beta_k$, and

$$\rho_k = [\alpha_k, \gamma_k]$$

Note the difference with (131)! The scalar product $\alpha_k^T\varphi$ is constant in the subspace of $R^d$ that is perpendicular to the scaling vector $\alpha_k$. Hence the function $g_k(\varphi)$ varies like $\sigma$ in a direction parallel to $\alpha_k$ and is constant across this direction. This motivates the term *semi-global* or *ridge-type* for this choice of functions.

As in (131) we expanded in the last equality in (134) the vector $\varphi$ with the value "1", again just to emphasize that the argument of the fundamental basis function $\sigma$ is bilinear in $\rho$ and $\varphi$.

### Connection to "Named Structures"

Here we briefly review some popular structures, other structures related to interpolation techniques are discussed in [Sjöberg et al., 1995, Juditsky et al., 1995].

**Wavelets** The local approach corresponding to (129,131) has direct connections to wavelet networks and wavelet transforms. The exact relationships are discussed in [Sjöberg et al., 1995]. Loosely, we note that via the dilation parameters in $\rho_k$ we can work with different scales simultaneously to pick up both local and not-so-local variations. With appropriate translations and dilations of a single suitably chosen function $\kappa$ (the "mother wavelet"), we can make the expansion (129) orthonormal. This is discussed extensively in [Juditsky et al., 1995].

**Wavelet and Radial Basis Networks.** The choice (133) without any orthogonalization is found in both wavelet networks, [Zhang and Benveniste, 1992] and radial basis neural networks [Poggio and Girosi, 1990].

**Neural Networks** The ridge choice (134) with

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

gives a much-used neural network structure, viz. the *one hidden layer feed-forward sigmoidal net.*

**Hinging Hyperplanes** If instead of using the sigmoid $\sigma$ function we choose "V-shaped" functions (in the form of a higher-dimensional "open book") Breiman's *hinging hyperplane* structure is obtained, [Breiman, 1993]. Hinging hyperplanes model structures [Breiman, 1993] have the form

$$g(x) = \max\left\{\beta^+ x + \gamma^+ \ , \ \beta^- x + \gamma^-\right\} \quad \text{or}$$
$$g(x) = \min\left\{\beta^+ x + \gamma^+ \ , \ \beta^- x + \gamma^-\right\} \ .$$

It can be written in a different way:

$$g(x) \ = \ \frac{1}{2}[(\beta^+ + \beta^-)x + \gamma^+ + \gamma^-] \pm \frac{1}{2}|(\beta^+ - \beta^-)x + \gamma^+ - \gamma^-| \ .$$

Thus a hinge is the superposition of a linear map and a semi-global function. Therefore, we consider *hinge* functions as semi-global or ridge-type, though it is not in strict accordance with our definition.

**Nearest Neighbors or Interpolation** By selecting $\kappa$ as in (130) and the location and scale vector $\rho_k$ in the structure (131), such that exactly one observation falls into each "cube", the nearest neighbor model is obtained: just load the input-output record into a table, and, for a given $\varphi$, pick the pair $(\widehat{y}, \widehat{\varphi})$ for $\widehat{\varphi}$ closest to the given $\varphi$, $\widehat{y}$ is the desired output estimate. If one replaces (130) by a smoother function and allow some overlapping of the basis functions, we get interpolation type techniques such as kernel estimators.

**Fuzzy Models** Also so called *fuzzy models* based on fuzzy set membership belong to the model structures of the class (129). The basis functions $g_k$ then are constructed from the fuzzy set membership functions and the inference rules. The exact relationship is described in [Sjöberg et al., 1995].

## 6.3 Estimating Non-linear Black Box Models

The model structure is determined by the following choices

- The regression vector (typically built up from past inputs and outputs)
- The basic function $\kappa$ (local) or $\sigma$ (ridge)
- The number of elements (nodes) in the expansion (129).

Once these choices have been made $\hat{y}(t|\theta) = g(\varphi(t), \theta)$ is a well defined function of past data and the parameters $\theta$. The parameters are made up of coordinates in the expansion (129), and from location and scale parameters in the different basis functions.

All the algorithms and analytical results of Section 2 can thus be applied. For Neural Network applications these are also the typical estimation algorithms used, often complemented with *regularization*, which means that a term is added to the criterion (24), that penalizes the norm of $\theta$. This will reduce the variance of the model, in that "spurious" parameters are not allowed to take on large, and mostly random values. See e.g. [Sjöberg et al., 1995].

For wavelet applications it is common to distinguish between those parameters that enter linearly in $\hat{y}(t|\theta)$ (i.e. the coordinates in the function expansion) and those that enter non-linearly (i.e. the location and scale parameters). Often the latter are seeded to fixed values and the coordinates are estimated by the linear least squares method. Basis functions that give a small contribution to the fit (corresponding to non-useful values of the scale and location parameters) can them be trimmed away ("pruning" or "shrinking").

# 7 User's Issues

## 7.1 Experiment Design

It is desirable to affect the conditions under which the data are collected. The objective with such *experiment design* is to make the collected data set $Z^N$ as informative as possible with respect to the models to be built using the data. A considerable amount of theory around this topic can be developed and we shall here just review some basic points.

The first and most important point is the following one

1. *The input signal $u$ must be such that it exposes all the relevant properties of the system.* It must thus not be too "simple". For example, a pure sinusoid

   $$u(t) = A \cos \omega t$$

   will only give information about the system's frequency response at frequency $\omega$. This can also be seen from (68). The rule is that

   - the input must contain at least as many different frequencies as the order of the linear model to be built.

     To be on the safe side, a good choice is to let the input be random (such as filtered white noise). It then contains all frequencies.

   Another case where the input is too simple is when it is generated by feedback such as

   $$u(t) = -Ky(t) \tag{135}$$

   If we would like to build a first order ARX model

   $$y(t) + ay(t-1) = bu(t-1) + e(t)$$

   we find that for any given $\alpha$ all models such that

   $$a + bK = \alpha$$

will give identical input-output data. We can thus not distinguish between these models using an experiment with (135). That is, we can not distinguish between any combinations of "$a$" and "$b$" if they satisfy the above condition for a given "$\alpha$". The rule is

- If closed-loop experiments have to be performed, the feedback law must not be too simple. It is to be preferred that a set-point in the regulator is being changed in a random fashion.

The second main point in experimental design is

2. *Allocate the input power to those frequency bands where a good model in particularly important.*

   This is also seen from the expression (68).

   If we let the input be filtered white noise, this gives information how to choose the filter. In the time domain it is often useful to think like this:

   - Use binary (two-level) inputs if linear models are to be built: This gives maximal variance for amplitude-constrained inputs.
   - Check that the changes between the levels are such that the input occasionally stays on one level so long that a step response from the system has time, more or less, to settle. There is no need to let the input signal switch so quickly back and forth that no response in the output is clearly visible.

   Note that the second point is really just a reformulation in the time domain of the basic frequency domain advice: let the input energy be concentrated in the important frequency bands.

   A third basic piece of advice about experiment design concerns the choice of sampling interval.

3. *A typical good sampling frequency is 10 times the bandwidth of the system.* That corresponds roughly to 5 – 7 samples along the rise time of a step response.

## 7.2 Model Validation and Model Selection

The system identification process has, as we have seen, these basic ingredients

- The set of models
- The data
- The selection criterion

Once these have been decided upon, we have, at least implicitly, defined a model: The one in the set that best describes the data according to the criterion. It is thus, in a sense, the best available model in the chosen set. But is it good enough? It is the objective of *model validation* to answer that question. Often the answer turns out to be "no", and we then have to go back and review the choice of model set, or perhaps modify the data set. See Figure 2!

How do we check the quality of a model? The prime method is to investigate how well it is capable of reproducing the behavior of a new set of data *(the validation data)* that was not used to fit the model. That is, we simulate the obtained model with a new input and compare this simulated output. One may then use one's eyes or numerical measurements of fit to decide if the fit in question is good enough. Suppose we have obtained several different models in different model structures (say a 4th order ARX model, a 2nd order BJ model, a physically parameterized one and so on) and would like to know which one is best. The simplest and most pragmatic approach to this problem is then to simulate each one of them on validation data, evaluate their performance, and pick the one that shows the most favorable fit to measured data. (This could indeed be a subjective criterion!)

The second basic method for model validation is to examine the residuals ("the leftovers") from the identification process. These are the prediction errors

$$\varepsilon(t) = \varepsilon(t, \hat{\theta}_N) = y(t) - \hat{y}(t|\hat{\theta}_N)$$

i.e. what the model could not "explain". Ideally these should be independent of information that was at hand at time $t-1$. For example if $\varepsilon(t)$ and $u(t-\tau)$

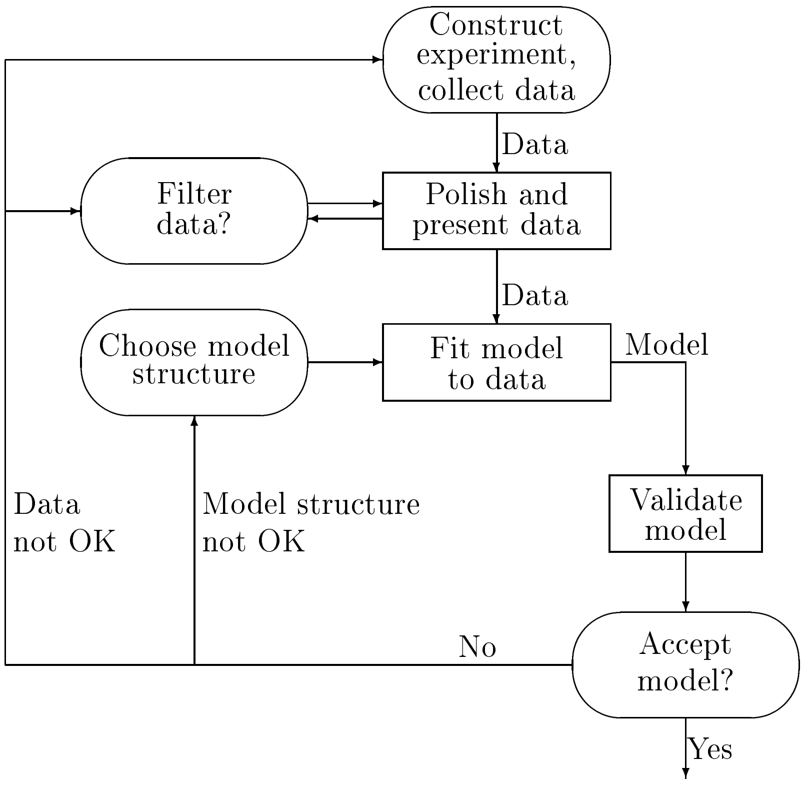

Figure 2: The identification loop

turn out to be correlated, then there are things in $y(t)$ that originate from $u(t-\tau)$ but have not been properly accounted for by $\hat{y}(t|\hat{\theta}_N)$ The model has then not squeezed out all relevant information about the system from the data.

It is good practice to always check the residuals for such (and other) dependencies. This is known as *residual analysis.*

## 7.3 Software for System Identification

In practice System Identification is characterized by some quite heavy numerical calculations to determine the best model in each given class of models. This is mixed with several user choices, trying different model structures, filtering data and so on. In practical applications we will thus need good software support. There are now many different commercial packages for identification available, such as Mathwork's System Identification Toolbox [Ljung, 1986], Matrix$_{\rm x}'$s System Identification Module [$MATRIX_x$, 1991] and PIM [Landau, 1990]. They all have in common that they offer the following routines:

**A** *Handling of data, plotting, etc.*

Filtering of data, removal of drift, choice of data segments, etc.

**B** *Non-parametric identification methods*

Estimation of covariances, Fourier transforms, correlation- and spectral-analysis, etc.

**C** *Parametric estimation methods*

Calculation of parametric estimates in different model structures.

**D** *Presentation of models*

Simulation of models, estimation and plotting of poles and zeros, computation of frequency functions, and plotting Bode diagrams, etc.

**E** *Model validation*

Computation and analysis of residuals ($\varepsilon(t, \hat{\theta}_N)$). Comparison between different models' properties, etc.

The existing program packages differ mainly in various user interfaces and by different options regarding the choice of model structure according to C above. For example, MATLAB's Identification Toolbox [Ljung, 1986] covers all linear model structures discussed here, including arbitrarily parameterized linear models in continuous time.

Regarding the user interface, there is now a clear trend to make it graphically oriented. This avoids syntax problems and relies more on "click and move", at the same time as tedious menu-labyrinths are avoided. More aspects of CAD tools for system identification are treated in [Ljung, 1993].

## 7.4 The Practical Side of System Identification

It follows from our discussion that the most essential element in the process of identification – once the data have been recorded – is to try out various model structures, compute the best model in the structures, using (24), and then validate this model. Typically this has to be repeated with quite a few different structures before a satisfactory model can be found.

The difficulties of this process should not be underestimated, and it will require substantial experience to master it. Here follows however a procedure that could prove useful to try out.

**Step 1: Looking at the Data**

Plot the data. Look at them carefully. Try to see the dynamics with your own eyes. Can you see the effects in the outputs of the changes in the input? Can you see nonlinear effects, like different responses at different levels, or different responses to a step up and a step down? Are there portions of the data that appear to be "messy" or carry no information. Use this insight to select portions of the data for estimation and validation purposes.

Do physical levels play a role in your model? If not, detrend the data by removing their mean values. The models will then describe how changes in the input give changes in output, but not explain the actual levels of the signals. This is the normal situation. The default situation, with good data, is that you detrend by removing means, and then select the first two thirds or so of the data record for estimation purposes, and use the remaining data for validation. (All of this corresponds to the "Data Quickstart" in the MATLAB Identification Toolbox.)

**Step 2: Getting a Feel for the Difficulties.**
Compute and display the spectral analysis frequency response estimate, the correlation analysis impulse response estimate as well as a fourth order ARX model with a delay estimated from the correlation analysis and a default order state-space model computed by a subspace method. (All of this corresponds to the "Estimate Quickstart" in the MATLAB Identification Toolbox.) This gives three plots. Look at the agreement between the

- Spectral Analysis estimate and the ARX and state-space models' frequency functions.
- Correlation Analysis estimate and the ARX and state-space models' transient responses
- Measured Validation Data output and the ARX and state-space models' simulated outputs. We call this the *Model Output Plot.*

If these agreements are reasonable, the problem is not so difficult, and a relatively simple linear model will do a good job. Some fine tuning of model orders, and noise models have to be made and you can proceed to Step 4. Otherwise go to Step 3.

**Step 3: Examining the Difficulties**
There may be several reasons why the comparisons in Step 2 did not look good. This section discusses the most common ones, and how they can be handled:

- **Model Unstable:** The ARX or state-space model may turn out to be unstable, but could still be useful for control purposes. Then change to a 5- or 10-step ahead prediction instead of simulation in the Model Output Plot.

- **Feedback in Data:** If there is feedback from the output to the input, due to some regulator, then the spectral and correlations analysis estimates are not reliable. Discrepancies between these estimates and the ARX and state-space models can therefore be disregarded in this case. In residual analysis of the parametric models, feedback in data can also be visible as correlation between residuals and input for negative lags.
- **Noise Model:** If the state-space model is clearly better than the ARX model at reproducing the measured output this is an indication that the disturbances have a substantial influence, and it will be necessary to carefully model them.
- **Model Order:** If a fourth order model does not give a good Model Output plot, try eighth order. If the fit clearly improves, it follows that higher order models will be required, but that linear models could be sufficient.
- **Additional Inputs:** If the Model Output fit has not significantly improved by the tests so far, think over the physics of the application. Are there more signals that have been, or could be, measured that might influence the output? If so, include these among the inputs and try again a fourth order ARX model from all the inputs. (Note that the inputs need not at all be control signals, anything measurable, including disturbances, should be treated as inputs).
- **Nonlinear Effects:** If the fit between measured and model output is still bad, consider the physics of the application. Are there nonlinear effects in the system? In that case, form the nonlinearities from the measured data. This could be as simple as forming the product of voltage and current measurements, if you realize that it is the electrical power that is the driving stimulus in, say, a heating process, and temperature is the output. This is of course application dependent. It does not cost very much work, however, to form a number of additional inputs by reasonable nonlinear transformations of the measured ones, and just test if inclusion of them improves the fit. See Example 2.
- **Still Problems?** If none of these tests leads to a model that is able to reproduce the Validation Data reasonably well, the conclu-

sion might be that a sufficiently good model cannot be produced from the data. There may be many reasons for this. The most important one is that the data simply do not contain sufficient information, e.g., due to bad signal to noise ratios, large and non-stationary disturbances, varying system properties, etc. The reason may also be that the system has some quite complicated non-linearities, which cannot be realized on physical grounds. In such cases, nonlinear, black box models could be a solution. Among the most used models of this character are the Artificial Neural Networks (ANN). See Section 6.

Otherwise, use the insights on which inputs to use and which model orders to expect and proceed to Step 4.

**Step 4: Fine Tuning Orders and Noise Structures**
For real data there is no such thing as a "correct model structure." However, different structures can give quite different model quality. The only way to find this out is to try out a number of different structures and compare the properties of the obtained models. There are a few things to look for in these comparisons:

- **Fit Between Simulated and Measured Output** Look at the fit between the model's simulated output and the measured one for the Validation Data. Formally, you could pick that model, for which this number is the lowest. In practice, it is better to be more pragmatic, and also take into account the model complexity, and whether the important features of the output response are captured.
- **Residual Analysis Test** You should require of a good model, that the cross correlation function between residuals and input does not go significantly outside the confidence region. A clear peak at lag $k$ shows that the effect from input $u(t-k)$ on $y(t)$ is not properly described. A rule of thumb is that a slowly varying cross correlation function outside the confidence region is an indication of too few poles, while sharper peaks indicate too few zeros or wrong delays.

- **Pole Zero Cancelations** If the pole-zero plot (including confidence intervals) indicates pole-zero cancelations in the dynamics, this suggests that lower order models can be used. In particular, if it turns out that the order of ARX models has to be increased to get a good fit, but that pole-zero cancelations are indicated, then the extra poles are just introduced to describe the noise. Then try ARMAX, OE, or BJ model structures with an A or F polynomial of an order equal to that of the number of non-cancelled poles.

**What Model Structures Should be Tested?**
Well, you can spend any amount of time to check out a very large number of structures. It often takes just a few seconds to compute and evaluate a model in a certain structure, so that you should have a generous attitude to the testing. However, experience shows that when the basic properties of the system's behavior have been picked up, it is not much use to fine tune orders in absurdum just to improve the fit by fractions of percents. For ARX models and state-space models estimated by subspace methods there are also efficient algorithms for handling many model structures in parallel.

**Multivariable Systems**
Systems with many input signals and/or many output signals are called multivariable. Such systems are often more challenging to model. In particular systems with several outputs could be difficult. A basic reason for the difficulties is that the couplings between several inputs and outputs leads to more complex models: The structures involved are richer and more parameters will be required to obtain a good fit.

Generally speaking, it is preferable to work with state-space models in the multivariable case, since the model structure complexity is easier to deal with. It is essentially just a matter of choosing the model order.

**Working with Subsets of the Input Output Channels:** In the process of identifying good models of a system it is often useful to select subsets of the input and output channels. Partial models of the system's behavior will then be constructed. It might not, for example, be clear if all measured inputs have a significant influence on the outputs. That is most easily tested by removing an input channel from the data, building a model for how the output(s) depend on the remaining input channels, and checking if there is a significant deterioration in the model output's fit to the measured one. See also the discussion under Step 3 above. Generally speaking, the fit gets better when more inputs are included and worse when more outputs are included. To understand the latter fact, you should realize that a model that has to explain the behavior of several outputs has a tougher job than one that just must account for a single output. If you have difficulties to obtain good models for a multi-output system, it might thus be wise to model one output at a time, to find out which are the difficult ones to handle. Models that just are to be used for simulations could very well be built up from single-output models, for one output at a time. However, models for prediction and control will be able to produce better results if constructed for all outputs simultaneously. This follows from the fact that knowing the set of all previous output channels gives a better basis for prediction, than just knowing the past outputs in one channel.

**step 5: Accepting the model** The final step is to accept, at least for the time being, the model to be used for its intended application. Recall the answer to question 10 in the introduction: *No matter how good an estimated model looks on your screen, has only picked up a simple reflection of reality. Surprisingly often, however, this is sufficient for rational decision making.*

## References

[Akaike, 1974a] Akaike, H. (1974a). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, AC-19:716–723.

[Akaike, 1974b] Akaike, H. (1974b). Stochastic theory of minimal realization. *IEEE Transactions on Automatic Control*, AC-19:667–674.

[Åström and Bohlin, 1965] Åström, K. J. and Bohlin, T. (1965). Numerical identification of linear dynamic systems from normal operating records. In *IFAC Symposium on Self-Adaptive Systems*, Teddington, England.

[Box and Jenkins, 1970] Box, G. E. P. and Jenkins, D. R. (1970). *Time Series Analysis, Forcasting and Control.* Holden-Day, San Francisco.

[Breiman, 1993] Breiman, L. (1993). Hinging hyperplanes for regression, classification and function approximation. *IEEE Trans. Info. Theory*, 39:999–1013.

[Brillinger, 1981] Brillinger, D. (1981). *Time Series: Data Analysis and Theory.* Holden-Day, San Francisco.

[Dennis and Schnabel, 1983] Dennis, J. E. and Schnabel, R. B. (1983). *Numerical methods for unconstrained optimization and nonlinear equations.* Prentice-Hall.

[Draper and Smith, 1981] Draper, N. and Smith, H. (1981). *Applied Regression Analysis, 2nd ed.* Wiley, New York.

[Juditsky et al., 1995] Juditsky, A., Hjalmarsson, H., Benveniste, A., Deylon, B., Ljung, L., Sjöberg, J., and Zhang, Q. (1995). Nonlinear black-box modeling in system identification: Mathematical foundations. *Automatica*, 31.

[Kushner and Yang, 1993] Kushner, H. J. and Yang, J. (1993). Stochastic approximation with averaging of the iterates: Optimal asymptotic rate of convergence for general processes. *SIAM Journal of Control and Optimization*, 31(4):1045–1062.

[Landau, 1990] Landau, I. D. (1990). *System Identificaiton and Control Design Using P.I.M. + Software.* Prentice Hall, Engelwood Cliffs.

[Larimore, 1983] Larimore, W. (1983). System identification, reduced order filtering and modelling via canonical variate analysis. In *Proc 1983 American Control Conference*, San Francisco.

[Ljung, 1986] Ljung, L. (1986). *The System Identification Toolbox: The Manual.* The MathWorks Inc. 1st edition 1986, 4th edition 1994, Natick, MA.

[Ljung, 1987] Ljung, L. (1987). *System Identification - Theory for the User.* Prentice-Hall, Englewood Cliffs, N.J.

[Ljung, 1993] Ljung, L. (1993). Identification of linear systems. In Linkens, D. A., editor, *CAD for Control Systems.* Marcel Dekker, New York.

[Ljung and Glad, 1994] Ljung, L. and Glad, T. (1994). *Modeling of Dynamic Systems.* Prentice Hall, Englewood Cliffs.

[Ljung and Söderström, 1983] Ljung, L. and Söderström, T. (1983). *Theory and Practice of Recursive Identification.* MIT press, Cambridge, Mass.

[$MATRIX_x$, 1991] $MATRIX_x$ (1991). $MATRIX_x$ users guide. *Integrated Systems Inc.*, Santa Clara, CA.

[Overschee and DeMoor, 1994] Overschee, P. V. and DeMoor, B. (1994). N4sid: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 30:75–93.

[Poggio and Girosi, 1990] Poggio, T. and Girosi, F. (1990). Networks for approximation and learning. *Proc. of the IEEE*, 78:1481–1497.

[Polyak and Juditsky, 1992] Polyak, B. T. and Juditsky, A. B. (1992). Acceleration of stochastic approximation by averaging. *SIAIM J. Control Optim.*, 30:838–855.

[Rissanen, 1974] Rissanen, J. (1974). Basis of invariants and canonical forms for linear dynamic systems. *Automatica*, 10:175–182.

[Rissanen, 1978] Rissanen, J. (1978). Modelling by shortest data description. *Automatica*, 14:465–471.

[Schoukens and Pintelon, 1991] Schoukens, J. and Pintelon, R. (1991). *Identification of Linear Systems: A Practical Guideline to Accurate Modeling.* Pergamon Press, London (U.K.).

[Sjöberg et al., 1995] Sjöberg, J., Zhang, Q., Ljung, L., Benveniste, A., Deylon, B., Glorennec, P., Hjalmarsson, H., and Juditsky, A. (1995). Nonlinear black-box modeling in system identification: A unified overview. *Automatica*, 31.

[Söderström and Stoica, 1989] Söderström, T. and Stoica, P. (1989). *System Identification.* Prentice-Hall Int., London.

[Solbrand et al., 1985] Solbrand, G., Ahlen, A., and Ljung, L. (1985). Recursive methods for off-line identification. *International Journal of Control*, 41(1):177–191.

[Zhang and Benveniste, 1992] Zhang, Q. and Benveniste, A. (1992). Wavelet networks. *IEEE Trans Neural Networks*, 3:889–898.

/home/rt/ljung/papers/levine/levine.tex