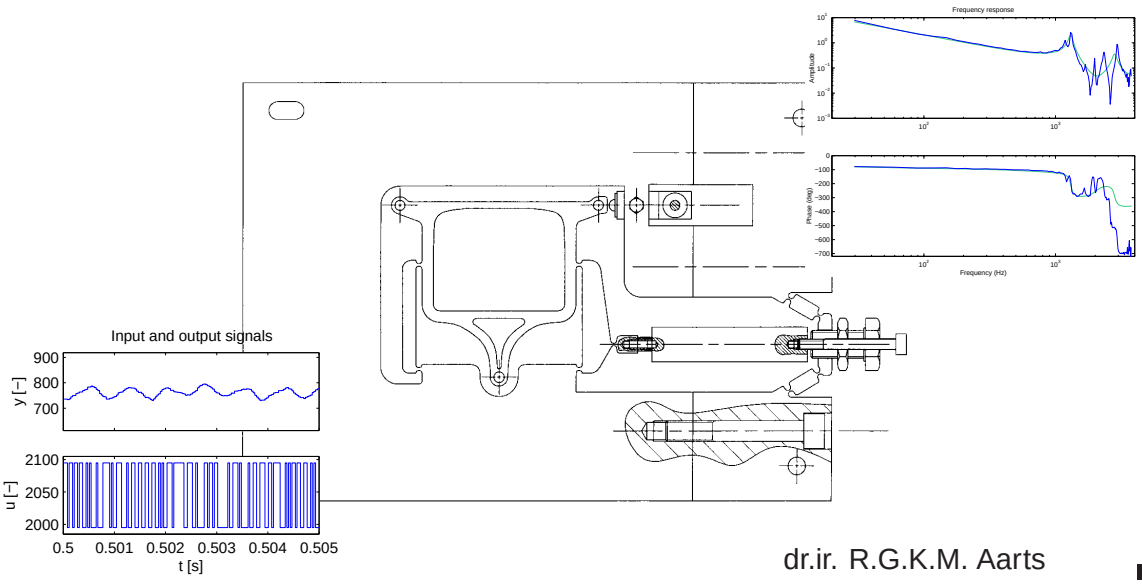
UNIVERSITY OF TWENTE.

Faculty of
Engineering Technology

Mechanical Automation and Mechatronics

System Identification and Parameter Estimation



dr.ir. R.G.K.M. Aarts

edition: 2011/2012

course: System Identification and
Parameter Estimation (191131700, ME / MA)

WA-nr.: **1232**

WA

© 1998–2012. Copyright R.G.K.M. Aarts, Universiteit Twente, Enschede.

All rights reserved. No part of this work may be reproduced, in any form or by any means, without the explicit and prior permission from the author.

Contents

List of symbols

Symbols:

A	System matrix in state space equations
B	Input matrix in state space equations
C	Output matrix in state space equations
D	Direct feedthrough matrix in state space equations
$\mathcal{E}$	Energy of a signal
G	Transfer function of the system
H	Transfer function of the noise signal
N	Number of samples
$\mathcal{P}$	Power of a signal
q	Forward shift operator
$R_x(\tau)$	(Auto)-covariance function of (stochastic) signal x
$R_{xy}(\tau)$	Cross-covariance function of signals x and y
s	Complex s plane (continuous-time systems)
T	Sample period
T_0	Total measuring time
u	System input
x	State vector
y	System output
z	Complex z plane (discrete-time systems)
Φ	Power Spectral Density function
Ψ	Energy Spectral Density function
ω	(Angular) frequency
ω_N	Nyquist frequency
ω_s	Sample frequency

Superscripts and subscripts:

G_0	Referring to the “true” system G

Notations:

$\|.\|$	Absolute value of a real number or the complex modulus (magnitude) of a complex number
$\angle.$	Argument (or phase) of a complex number.
$.^*$	Complex conjugate
$\mathrm{Im}(.)$	Imaginary part of complex number
$\mathrm{Re}(.)$	Real part of complex number
$\mathcal{F}\{.\}$	Fourier transform of a signal
$\mathcal{L}\{.\}$	Laplace transform of a signal
$\mathcal{Z}\{.\}$	z transform of a signal
Ex	Expectation of x
$\hat{G}$	Estimation of G

1 Introduction

There may be many reasons to look for a model of a system based on experimental data. In this introduction we will focus on two techniques that can be used to obtain such models: System Identification and Parameter Estimation. Although the goals and applications can be quite different, both techniques also share common issues.

With *System Identification* we use dedicated experiments to find a compact and accurate mathematical model of a dynamic system. It is frequently applied to optimise a controller for that system with knowledge from a sufficiently accurate description of the process. In many practical cases we have only a limited model of the dynamic behaviour of a complex process. In other cases the important parameters are only approximately known. In those cases System Identification can assist to obtain the required dynamic model. In this lecture we will discuss the design of the experiments and the data processing using MATLAB's [8] identification toolbox IDENT [7]. The techniques will e.g. be applied to analyse the dynamic behaviour of mechanical systems.

An important step in the identification procedure is the estimation of the parameters in the model. As will be discussed, System Identification will focus on a special class of systems and accompanying models. In these lecture notes the term *Parameter Estimation* will be used for more general cases, including non-linear systems. Furthermore, the models will have a clear physical background and hence the parameters involved have a physical meaning. For non-linear optimisation problems MATLAB's Optimization Toolbox [9] may be used.

1.1 What are System Identification and Parameter Estimation?

The goal of System Identification is a *mathematical* model of a *dynamic* system based on *experimental* data. The model should be compact and adequate with respect to the purpose it is to be used for. In the cases to be discussed here the latter usually means that it can be used to design a controller for the process.

In System Identification we are generally dealing with *mathematical* model as opposed to *physical* models. The differences are illustrated in the following table:

Physical models	vs.	**Mathematical models**
microscopic		macroscopic
"First principles": conservation laws, physical properties		Experimental: planned experiments, look for relations in the data, system identification
PDE's: ∞-dimensional		ODE's: finite dimension
non-linear		LTI
no general-purpose technique		"standard" techniques

As indicated we generally consider LTI systems, that are Linear Time Invariant systems. Where possible, non-linearities are removed from the data by preprocessing. LTI *dynamic* systems can be described with

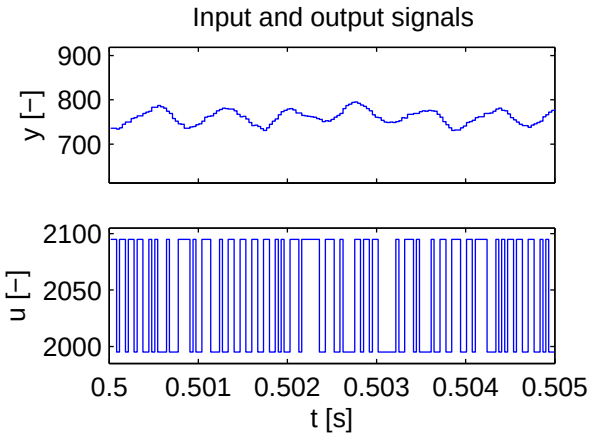


Figure 1.1: Example of input and corresponding output signal of a mechanical system.

a transfer function G that relates the input u and output y

$$\frac{y(s)}{u(s)} = G(s). \tag{1.1}$$

In this expression a continuous transfer function is represented in the complex s-plane. Remember that in general the models are mathematical. That means that no knowledge of the physical background is required. Where such knowledge is available it may be used e.g. to select model structure (mathematical structure, order of the transfer function).

The differences between mathematical and physical modelling can be illustrated with the transfer function of a single mass m that can move in one direction x (output). The motion is initiated by a force F (input) exerted on the mass. Furthermore, the mass is attached to the world with a spring, stiffness k, and a (viscous) damper, damping d. Then the transfer function of the system is [1]

$$G(s) = \frac{x}{F} = \frac{1}{ms^2 + ds + k}. \tag{1.2}$$

The parameters in this model clearly represent the physical properties of the system. On the other hand, a typical model used for System Identification might look like

$$G(s) = \frac{x}{F} = \frac{b_1}{s^2 + a_1 s + a_2}. \tag{1.3}$$

Here, physical insight is used to select a second order model, but the parameters do not need to have a clear physical meaning.[1]

The *experiments* are designed to obtain as much useful information of the system as possible. An input signal is supplied to the system and its response is measured. An example is shown in Fig. 1.1. This shows the response of a mechanical system which will be explained in more detail later. The input signal is a random binary signal. Such input and output signals are analysed, where in many cases we refer to the IDENT toolbox.

This toolbox has a user-friendly graphical interface as is illustrated in Fig. 1.2. In these lecture notes we will discuss the functions behind the buttons on the GUI and the mathematics behind the functions. Also suggestions on the procedures to follow will be given. A typical procedure includes:

- Input/output selection
- Experiment design
- Collecting the data
- Selection of the model structure (that defines the set of possible solutions)
- Estimation of the parameters
- Validation of the model (preferable with independent “validation” data).

[1] As will be discussed later, System Identification is often carried out with discrete time models.

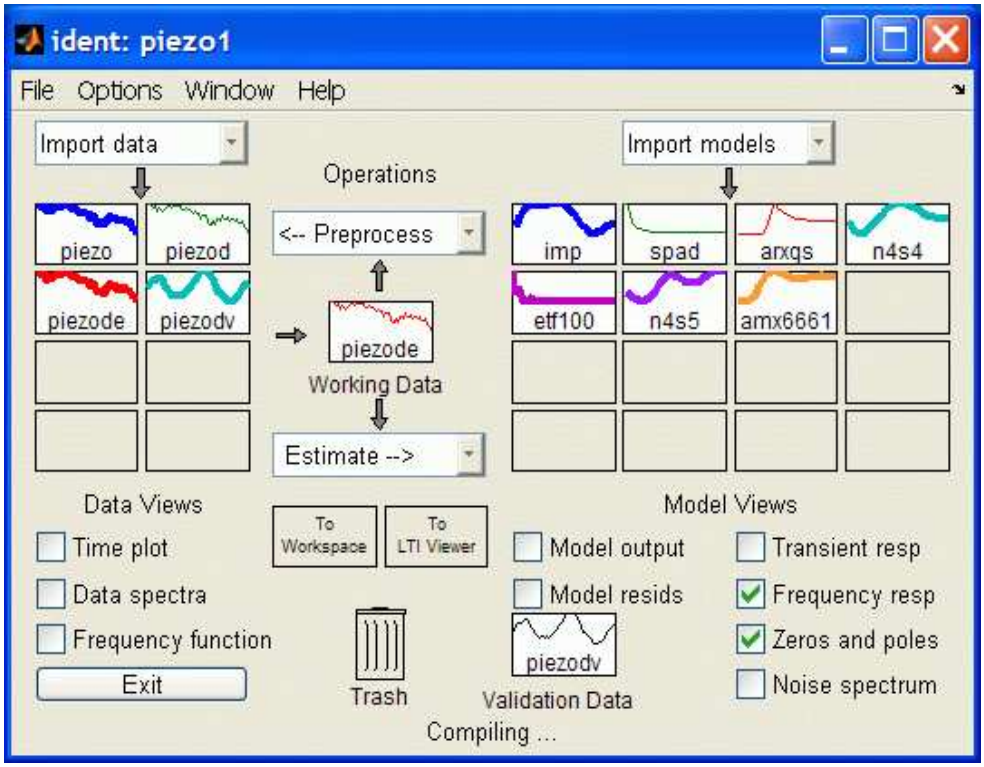


Figure 1.2: Example of IDENT's GUI.

In these lecture notes, Parameter Estimation will also be used to find models of dynamic systems from experimental data, but it will typically be used in combination with a *physical* model. Hence the obtained parameters should represent certain physical properties of the system being examined. The models may be linear as well as non-linear. It will be shown that the linearity of the model (or the system being examined) is not the main reason for the complexity of the parameter estimation. More precisely, it may be possible to derive a set of equations for a non-linear system that is *linear-in-the-parameters*. In that case, a straightforward procedure can be applied to uniquely determine the "best" parameter set. In reverse, even a linear system can easily give rise to equations that are non-linear-in-the-parameters and non-linear optimisation techniques have to be applied.

As outlined above, we will deal mostly with *dynamic* systems, which means that the time dependency of the signals plays an important role. It also means that the models describing the systems will include differential equations in which time derivatives (or their discrete time equivalent differences) are present. Nevertheless are the applied techniques closely related to curve fitting techniques that are commonly applied to match experimental data and non-dynamic models, i.e. models in which the time does not play a role. Examples will include such curve fitting as well.

1.2 Overview

Topics covered by these lecture notes can be distinguished into topics closer related to System Identification and topics on more general Parameter Estimation, although the differences are often smaller than may appear at first sight.

Before any identification of estimation technique is discussed, some general topics are discussed first. These include discrete-time dynamic systems, basic signal theory and stochastic theory and in introduction into time domain and frequency domain analyses.

Next System Identification techniques are outlined for open-loop LTI SISO systems. So-called non-

parametric identification uses correlations and spectral analysis to obtain estimates of the impulse response or frequency response function. Such results can provide valuable insight for subsequent parametric identification tools like subspace identification to estimate state space models. Transfer functions are estimated with Prediction error methods for which the prediction of the output from past data plays an important role. Topics to be discussed are the model structure, approximate models, order selection, validation and experiment design.

Similarly Parameter Estimation is applied to deal with non-linear model equations. In the case the model is linear in the parameters a straightforward linear optimisation procedure is applied to estimate the parameters. Not all parameters are necessarily identifiable. Identifiable parameters may be very sensitive to measurement noise. Again the design of experiments has to be considered.

Finally more “advanced” topics are addressed: MIMO-systems (for System Identification), identification in the frequency domain, identification in the presence of a control system (closed loop systems) and non-linear optimisation. A number of cases illustrate the use of the discussed techniques.

2 Systems and signals

2.1 Continuous-time systems

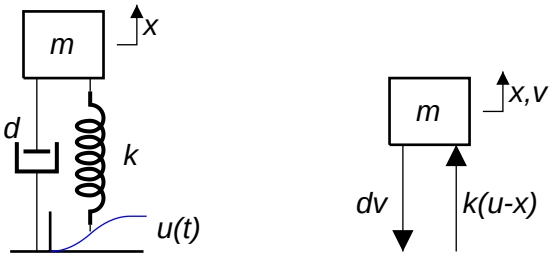
In an analysis for control purposes dynamic models of "real" systems are often written as *continuous-time* models consisting of a set of ordinary differential equations (ODE's). E.g. the equation of motion for the position x of a single mass m attached to one side of a spring with stiffness k that is excited with amplitude u at the other side is

$$m\ddot{x}(t) = -k\left(x(t) - u(t)\right). \tag{2.1}$$

If we consider also viscous damping for the motion of the mass, see Fig. 2.1, an extra term is added to this equation and we write

$$m\ddot{x}(t) = -k\left(x(t) - u(t)\right) - d\dot{x}(t), \tag{2.2}$$

in which the parameter d represents the viscous damping coefficient.



(a) (b)

Figure 2.1: Mass with viscous damping and set into motion via a spring: (a) Schematic overview and (b) forces working on the mass.

Considering the Laplace transforms of a function $f(t)$

$$\mathcal{L}\{f(t)\} = F(s) = \int_0^\infty f(t)e^{-st}\mathrm{d}t \tag{2.3}$$

an important property is (with zero initial conditions)

$$\mathcal{L}\{\dot{f}(t)\} = sF(s). \tag{2.4}$$

Taking the Laplace transform of x and u in Eq. (2.2), we write in the complex s-domain

$$(ms^2 + ds + k)X(s) = kU(s), \tag{2.5}$$

or, after defining the transfer function $G(s)$

$$G(s) = \frac{X(s)}{U(s)} = \frac{k/m}{s^2 + d/m\, s + k/m}. \tag{2.6}$$

Note that this expression is often written without taking the Laplace transforms, so

$$G(s) = \frac{x(t)}{u(t)} = \frac{k/m}{s^2 + d/m\, s + k/m}, \tag{2.7}$$

which implies an association of s with an s-operator that computes the derivative of a function.

Next, we consider an input signal $u(t)$ that is a harmonic signal with angular frequency ω

$$u(t) = u_0 \sin(\omega t). \tag{2.8}$$

If the system is properly damped, then after some time the transient behaviour of the system will damp and the output $x(t)$ is also harmonic with the same frequency and its amplitude and phase with respect to $u(t)$ are determined by the (complex) value of $G(i\omega)$, that is the (complex) number that is obtained when $s = i\omega$ is substituted in the expression of the transfer function $G(s)$. More precisely, the gain $|G(i\omega)|$ equals the ratio of the amplitudes of output and input signals and the phase angle $\angle G(i\omega)$ is equal to the phase shift. The gain and phase shift are shown as functions of the (angular) frequency in a Bode plot or Frequency Response Function (FRF), see Fig. 2.2.

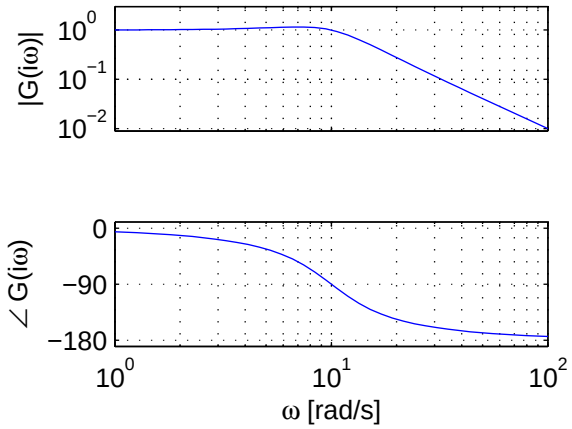


Figure 2.2: Typical Bode plot or Frequency Response Function (FRF) for a system as shown in figure 2.1.

The information provided by such a plot is basically a model of the LTI system as it can, in principle, be used to compute the output of the system for a given input. First we need to determine which frequencies are present in the input. A signal expressed in the time domain is transformed into the frequency domain with a Fourier transform as will discussed in Sect. 2.3. Next for each frequency the gain and phase shift are determined from the FRF so the output signal can be computed in the frequency domain. Finally, a time domain signal is obtained with an inverse Fourier transform.

Alternatively, the system's behaviour can be expressed directly in the time domain by means of the impulse response $g(t)$, see Fig. 2.3. This impulse response equals the output of the system when it is excited by an (unit) impulse $\delta(0)$ at $t = 0$, that is an infinitely short signal, so

$$\delta(t) = 0 \quad \text{for} \quad t \neq 0, \tag{2.9}$$

which satisfies

$$\int_{-\infty}^{\infty} f(t)\delta(t)\, dt = f(0) \tag{2.10}$$

for any function $f(t)$ that is continuous in $t = 0$. Then any input signal $u(t)$ can be written as

$$u(t) = \int_0^{\infty} \delta(\tau)u(t-\tau)\, \mathrm{d}\tau, \tag{2.11}$$

so as a continuous series of impulses at every time instant. For each of these impulses the output will show the impulse response. As the system is linear, all these responses can be added or, stated mathematically more precise, can be integrated according to

$$y(t) = (g * u)(t) = \int_0^\infty g(\tau)u(t-\tau)\,\mathrm{d}\tau, \tag{2.12}$$

to compute the output $y(t)$ for the specified input signal $u(t)$. The notation "$(g * u)$" is used to denote the *convolution* of two signals which is actually defined in this equation. Apparently, also the impulse response $g(t)$ provides, in principle, a model of a system that includes all necessary information to compute the output $y(t)$ for a given input $u(t)$.

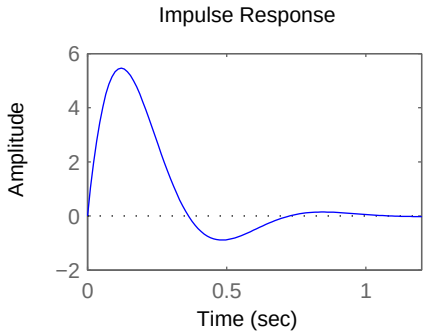


Figure 2.3: Typical impulse response for a system as shown in figure 2.1.

Note that the poles of $G(s)$, which are the values of s where the denominator of the transfer function $G(s)$ equals zero, are very important for the dynamic behaviour of the system. E.g. for stability all poles should be in the left half plane of the complex plane, i.e. the real part should obey $\mathrm{Re}\, s < 0$.

Continuous-time systems can be identified by some techniques like e.g. in the FREQID toolbox by de Callafon and van der Hof [2]. In most cases however, the identification will deal with discrete-time systems as discussed in the next section. This may seem somewhat "unphysical" at first sight, but one should not forget that in an identification experiment usually digital equipment is used to excite a system and to measure the response. Then actually the complete (discrete-time) system consisting of the "physical" (continuous) system to be studied and the D/A and A/D converters is identified. In case the identification is for the purpose of (discrete-time) controller design, this discrete-time system is actually the system the controller has to deal with and therefore is the most obvious system to estimate.

2.2 Discrete-time systems

In discrete-time systems we consider *digitised* signals. That are signals $x(k)$ (or alternatively written as x_k) defined only at discrete time steps $t_k = kT$ where T is the sample period and k is an integer, see Fig. 2.4. In between the time steps the signal is in general not defined. That is no problem as long as one discrete-time system is connected to another discrete-time system with matching sample times.

However, as soon as a discrete-time system is connected to a continuous-time system, it has to be specified somehow what signal values are supplied to the continuous-time system at all time instances. In many cases it is assumed that a signal x is set to a value $x(k)$ at time t_k and remains at that value until the next time step t_{k+1} (zero-order hold or ZOH discretisation).

Furthermore, when the output of a continuous-time system is digitised, then we would like to be able to describe the dynamic properties of the system with the discrete-time samples of the output signal. Most likely, the continuous-time model of the system will involve differential equations with time

(a) (b)

Figure 2.4: Example of (a) a continuous-time signal and (b) a discrete-time signal.

derivatives like $\dot{x}(t)$ and higher order derivatives which are well-defined for continuous-time signals $x(t)$, but have no meaning for discrete-time signals $x(k)$.

In stead, we will use *difference* equations where changes in a signal like $x(k+1) - x(k)$ are considered. The correspondence with differential equations can be illustrated by considering e.g. Euler's forward approximation of the first derivative of a continuous signal x that is sampled at discrete time steps, namely

$$\dot{x}(t = kT) \approx \frac{x(k+1) - x(k)}{T}. \tag{2.13}$$

Many textbooks give introductions on the dynamic analysis of discrete systems, see e.g. Franklin *et al.* [3, Chapter 8]. In these notes some of the notations and properties of these systems will be given.

Introducing the forward and backward shift operators q and q^{-1}, respectively,

$$\begin{aligned} qx(k) &= u(k+1), \\ q^{-1}x(k) &= u(k-1), \end{aligned} \tag{2.14}$$

Eq. (2.13) can be written shorter as

$$\dot{x} \approx \frac{(q-1)x(k)}{T}. \tag{2.15}$$

Similar to the Laplace transform for continuous systems the *z-transform* is defined for discrete signals

$$\mathcal{Z}\{f(k)\} = F(z) = \sum_{k=0}^{\infty} f(k)z^{-k}. \tag{2.16}$$

Then obviously

$$\mathcal{Z}\{f(k-1)\} = z^{-1}\mathcal{Z}\{f(k)\}, \tag{2.17}$$

or using the shift operator

$$\mathcal{Z}\{q^{-1}f(k)\} = z^{-1}\mathcal{Z}\{f(k)\}. \tag{2.18}$$

Then it is also possible to write a relation between two signals like

$$x(k) = -a_1 x(k-1) + b_1 u(k-1) + b_2 u(k-2) \tag{2.19}$$

with a transfer function in the z-plane after applying the z-transform

$$G(z) = \frac{x(z)}{u(z)} = \frac{b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1}}, \tag{2.20}$$

or alternatively using the q-operator

$$G(q) = \frac{x(k)}{u(k)} = \frac{b_1 q^{-1} + b_2 q^{-2}}{1 + a_1 q^{-1}}. \tag{2.21}$$

Following Ljung [6, Sect. 2.1] we will mostly use the second notation.

An important property of the discrete transfer function $G(q)$ is that "equivalent characteristics in the z-plane are related to those in the s-plane by the expression

$$z = e^{sT}, \tag{2.22}$$

where T is the sample period" [3, Page 655].

The importance of this property can be illustrated by considering the stability border for the poles of a system. For a stable continuous-time system it was pointed out the the poles have to satisfy $\mathrm{Re}\, s < 0$. Obviously, the imaginary axis is the border of stability and the poles have to be in the left half plane (LHP). According to Eq. (2.22) for discrete-time systems the role of the imaginary axis is taken over by the unit circle. E.g. the poles of a stable discrete-time system have to satisfy

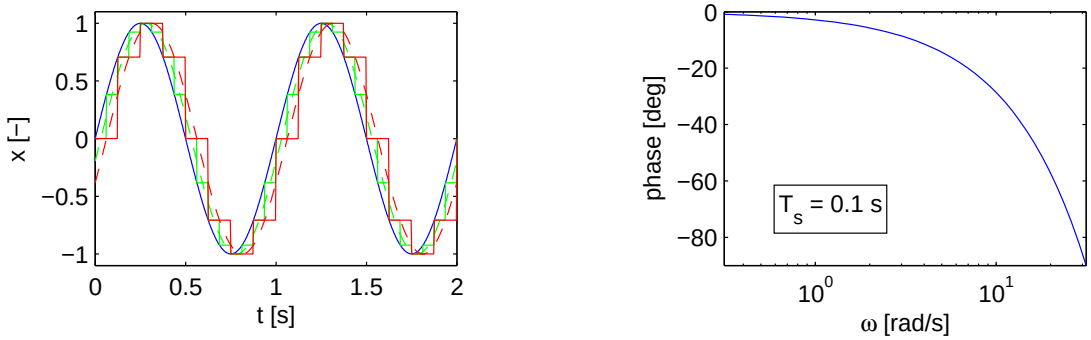
$$|z| < 1, \tag{2.23}$$

which means that they have to be inside the unit circle.

Furthermore, the amplification and phase shift of a harmonic output signal with respect to the input signal is given by the (complex) number $G(e^{i\omega T})$, where ω is the angular frequency considered. In other words,

$$q = e^{i\omega T} \tag{2.24}$$

is substituted into the discrete-time transfer function $G(q)$ to obtain the frequency domain response of the system with gain $|G(e^{i\omega T})|$ and phase shift $\angle G(e^{i\omega T})$.



(a) (b)

Figure 2.5: The phase lag introduced by ZOH discretisation: (a) illustrated for a harmonic signal, (b) frequency dependence of the phase lag.

As stated above, discretisation of a continuous time signal plays an important role when e.g. the output of a continuous physical system is measured by a digital computer that only samples at the discrete time steps. On a continuous time scale the discrete time signals are commonly represented using zero-order hold (or ZOH) discretisation, where the sampled signal u_k at time t_k is kept constant until the next sample u_{k+1} is measured at time t_{k+1}. Figure 2.5(a) illustrates this for a harmonic signal (sine function) and *sample frequencies* of $f_s = 8$ Hz and $f_s = 16$ Hz respectively. It shows that the sampled signal has a *phase lag*. That means that the sampled signal appears to lag behind the original continuous time signal. This is illustrated by the dashed curves in Fig. 2.5(a). These dashed lines represent a reconstruction of the time continuous signal that is obtained by matching the respective ZOH signal as closely as possible. Obviously, this reconstructed signal lags one half sample period, so $\frac{1}{2}T$, behind the original signal.

This time delay can be represented by a phase delay that depends on the (angular) frequency ω of the original signal. For "slow" signals, the difference between original and ZOH discretised signal will

be hardly noticeable. This is quite different near the Nyquist (angular) frequency ω_{N}, which is half the sample (angular) frequency ω_s:

$$\omega_{\mathrm{N}} = \frac{1}{2}\omega_s. \tag{2.25}$$

At this frequency the phase lag is -90^{o}. Figure 2.5(b) shows the phase lag as a function of the signal frequency ω up to the Nyquist frequency. It points out once more that the phase lag introduced in experimental data due to sampling, has to be taken into account for frequencies that are too close to the Nyquist frequency, so half of the sampling frequency.

2.3 Fourier transforms

In signal theory the frequency domain plays an important role. In the previous sections the behaviour of a harmonic signal of some (angular) frequency ω was shortly mentioned. More generally, we are interested in the behaviour of a system as a function of the frequency as e.g. is expressed by $G(e^{i\omega T})$.

2.3.1 Signal types

It will turn out that different *types* of signals need to be analysed differently. Figure 2.6 illustrates the three signal types we will consider: deterministic signals with (a) finite energy or (b) finite power and (c) stochastic signals.

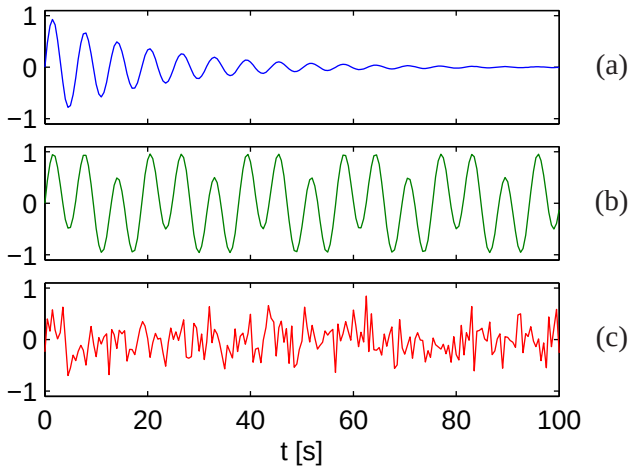


Figure 2.6: Signal types: (a) finite energy, (b) finite power, and (c) a realization of a stationary stochastic process.

In order to make this signal classification, some quantities will be introduced:

- The frequency content of a signal in the time is obtained by applying a Fourier transform. It indicates the frequencies that are present in a signal. More specific, the outcome of the Fourier transform are the amplitudes and the phases of those frequencies.
- The energy content of a signal can also be considered as a function of the frequency.
- The same holds true for the distribution of the power content over the frequencies.

Deterministic signal will be discussed first, followed by some additional remarks for stochastic signals.

2.3.2 Deterministic signals

Fourier transforms relate signals in the time domain with their description in the frequency domain and vice versa. Mathematically this transform is defined by means of Fourier integrals for infinite-time continuous-time signals. This case will be discussed first. Next Fourier series are outlined for periodic signals and finite-time signals. Finally, the more practical *Discrete Fourier Transform* (DFT) will be addressed. This DFT will be used most of the time to analyse finite-time discrete-time signals.

Infinite-time, continuous-time signals

Mathematically the Fourier transforms are defined by a pair of *Fourier integrals*

$$U(\omega) = \int_{-\infty}^{\infty} u(t) e^{-i\omega t}\, \mathrm{d}t, \qquad u(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} U(\omega) e^{i\omega t}\, \mathrm{d}\omega, \tag{2.26}$$

where $U(\omega)$ is the Fourier transform of the signal $u(t)$. The integrals in Eq. (2.26) do not exist for all signals $u(t)$.

The Fourier integrals can be evaluated for so-called *energy signals*. That are signals with a finite energy, i.e.

$$\int_{-\infty}^{\infty} u(t)^2 \mathrm{d}t < \infty, \tag{2.27}$$

see e.g. signal (a) in Fig. 2.6. As illustrated in this figure, the absolute value of such a signal with a finite energy will go to zero, $|u(t)| \to 0$ for $t \to \infty$.

The Fourier integrals can also be computed for deterministic signals with finite power (*power signals*), i.e.

$$\lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} u(t)^2 \mathrm{d}t < \infty, \tag{2.28}$$

see e.g. signal (b) in Fig. 2.6. For these signals the Fourier transform will in general be unbounded as the energy is unbounded. Dirac impulses (the δ-function according to Eq. (2.10)) are then used to represent the frequency spectrum.

Finite-time, continuous-time signals

In case a signal is only defined in a finite interval $[0, T_0]$ there are two possibilities to analyze them using Eq. (2.26). One way is to assume the signal is zero outside the interval. Then Eq. (2.26) can be written as

$$U_{T_0}(\omega) = \int_{0}^{T_0} u(t) e^{-i\omega t}\, \mathrm{d}t, \qquad u(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} U(\omega) e^{i\omega t}\, \mathrm{d}\omega. \tag{2.29}$$

Alternatively, the signal can be extended periodically outside the interval $[0, T_0]$ with period T_0. Then the Fourier transform is not a continuous function anymore, but is a *Fourier series* consisting of the *Fourier coefficients* c_l

$$c_l = \frac{1}{T_0} U_{T_0}(l\omega_0) = \frac{1}{T_0} \int_{0}^{T_0} u(t) e^{-il\omega_0 t}\, \mathrm{d}t, \qquad u(t) = \sum_{l=-\infty}^{\infty} c_l e^{il\omega_0 t}, \tag{2.30}$$

with

$$\omega_0 = \frac{2\pi}{T_0}. \tag{2.31}$$

Infinite-time, discrete-time signals

For a description of the Fourier transform of a discrete-time signal u_k, we first try to analyse such a signal with the Fourier transforms introduced for continuous-time signals. Of course, the discrete-time signal u_k is only defined at the discrete time instances $t = kT$ where T is the sample period, e.g. by sampling a continuous-time signal $u(t)$ at these time instances

$$u_k = u(kT), \quad k = ..., -2, -1, 0, 1, 2, ... \tag{2.32}$$

From the discrete-time signal u_k we will reconstruct a (mathematically) continuous-time signal u_s consisting of a series of impulses

$$u_s(t) = \sum_{k=-\infty}^{\infty} u_k \delta(t - kT). \tag{2.33}$$

This reconstructed signal $u_s(t)$ is related to the original continuous-time signal $u(t)$ according to

$$u_s(t) = \sum_{k=-\infty}^{\infty} u(t) \delta(t - kT). \tag{2.34}$$

Next we consider the Fourier transform of $u_s(t)$ which is according to Eq. (2.26)

$$U_s(\omega) = \int_{-\infty}^{\infty} u_s(t) e^{-i\omega t} \, \mathrm{d}t. \tag{2.35}$$

After substitution of Eq. (2.34) this can be written as

$$U_s(\omega) = \sum_{k=-\infty}^{\infty} u(kT) e^{-i\omega kT}. \tag{2.36}$$

One immediate conclusion from this expression is that the Fourier transform $U_s(\omega)$ is a periodic function of the (angular) frequency

$$U_s(\omega) = U_s(\omega + \omega_s), \tag{2.37}$$

in which the period along the frequency axis equals the sample frequency

$$\omega_s = \frac{2\pi}{T}. \tag{2.38}$$

Furthermore, using properties of convolutions (see page 2-13) it can be proven that the Fourier transform of the reconstructed signal $U_s(\omega)$ is related to the Fourier transform of the original signal $U(\omega)$ according to

$$U_s(\omega) = U(\omega) * \mathcal{F}\left\{ \sum_{k=-\infty}^{\infty} \delta(t - kT) \right\}, \tag{2.39}$$

in which the "$*$" indicates a convolution and $\mathcal{F}\{.\}$ denotes the Fourier transform of a signal. As the series of δ-functions represents a periodic signal, its Fourier transform is a series that can be written as

$$\mathcal{F}\left\{ \sum_{k=-\infty}^{\infty} \delta(t - kT) \right\} = \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} \delta(\omega - \frac{2\pi k}{T}). \tag{2.40}$$

Then we can write

$$U_s(\omega) = U(\omega) * \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} \delta(\omega - \frac{2\pi k}{T}). \tag{2.41}$$

Substituting the sample frequency ω_s and evaluating the convolution yields

$$U_s(\omega) = \omega_s \int_{-\infty}^{\infty} U(\Omega) \sum_{k=-\infty}^{\infty} \delta(\omega - \Omega - \frac{2\pi k}{T}) \, \mathrm{d}\Omega, \tag{2.42}$$

so finally

$$U_s(\omega) = \omega_s \sum_{k=-\infty}^{\infty} U(\omega - k\omega_s). \tag{2.43}$$

Apparently, the Fourier transform $U_s(\omega)$ of the signal $u_s(t)$ that is reconstructed from the discrete-time signal u_k with only information of the original signal $u(t)$ at time instances $t = kT$, shows the following properties:

- The Fourier transform $U_s(\omega)$ is periodic with period ω_s, the sample frequency, Eq. (2.37).
- The Fourier transform $U_s(\omega)$ can be constructed with a summation of the Fourier transform of the original signal $U(\omega)$ in which the latter are shifted any multiple of the sample frequency ω_s, Eq. (2.43).

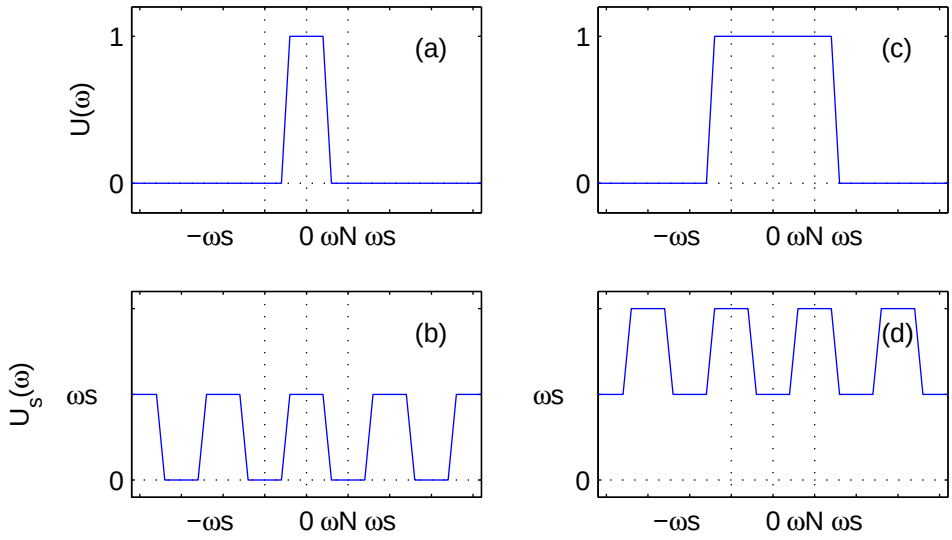


Figure 2.7: Fourier transforms of the original signal $U(\omega)$ and the reconstructed signal $U_s(\omega)$ for signals with frequencies only inside or also outside the range to the Nyquist frequency ω_{N}.

With these properties in mind there are two essentially different cases for the original signal $u(t)$: It consists of only frequencies within the frequency band op to the Nyquist frequency ω_{N}, Fig. 2.7(a), or it has frequency components above the Nyquist frequency ω_{N}, Fig. 2.7(c).

In the first case the Fourier transform of the reconstructed signal $U_s(\omega)$ in the range from $-\omega_{\mathrm{N}}$ to ω_{N} equals the original Fourier transform $U(\omega)$ multiplied with ω_s. Outside this interval $U_s(\omega)$ can be constructed easily because of the periodic nature, Fig. 2.7(b). Now suppose we have our discrete-time signal u_k and we would like to reconstruct the original continuous-time signal $u(t)$, then this appears to be possible, in principle: Compute the Fourier transform $U_s(\omega)$, apply low pass filtering to leave only frequencies up to the Nyquist frequency and divide the result by ω_s. Then we have computed $U(\omega)$ from which $u(t)$ can be reconstructed perfectly with an inverse Fourier transform.

In the second case, however, we notice immediately that $U(\omega)$ contributes to $U_s(\omega)$ also outside the range from $-\omega_{\mathrm{N}}$ to ω_{N}. Next when the periodic contributions are added according to Eq. (2.43),

we notice that for some frequencies lower and higher frequencies mix in the Fourier transform of the reconstructed signal $U_s(\omega)$, Fig. 2.7(d). As a result it is no longer possible to uniquely reconstruct the Fourier transform $U(\omega)$ from the discrete-time signal u_k!

What we observe is stated more precisely in the Nyquist-Shannon sampling theorem. The theorem considers the reconstruction of the continuous-time signal from the samples of a discrete-time signal that is obtained by sampling the original continuous time signal. It states that this reconstruction can only be carried out without loss of information provided that the sample frequency ω_s is at least twice the largest frequency present in the original signal. In other words, the highest frequency component in the original signal should be less than half the sampling rate, which is referred to as the Nyquist frequency ω_{N}.

In the case the signal does have frequency components that are above the Nyquist frequency as in Fig. 2.7(c), then after sampling these frequency components are shifted along the frequency axis by a multiple of the sample frequency ω_s, such that they will show up at a frequency inside the range from $-\omega_{\mathrm{N}}$ to ω_{N}. This phenomenon is called *aliasing* and is undesirable in most applications. It is essential to recognise that it occurs in the system where sampling takes place. If aliasing should be avoided then measures have to be taken *before* the signal is sampled, e.g. by using an electronic circuit for low-pass filtering.

From the Fourier transform outlined above we recall Eq. (2.36) with which the Fourier transform of the discrete-time signal $u_k = u(kT)$ can be computed. Note that it may be complicate or even impossible to reconstruct the continuous-time signal. Nevertheless, it is quite straightforward to apply the inverse transform to compute the discrete-time signal. The resulting expressions can be written as

$$U(\omega) = \sum_{k=-\infty}^{\infty} u_k e^{-ik\omega T}, \qquad u_k = \frac{T}{2\pi}\int_{2\pi/T} U(\omega) e^{ik\omega T}\,\mathrm{d}\omega. \tag{2.44}$$

Note that the subscript s is not used in these expressions, although $U(\omega)$ indicates the Fourier transform of the discrete-time signal u_k. Recall from Eq. (2.37) that the Fourier transform $U(\omega)$ is periodic with length equal to the sample frequency ω_s. That means that the integral of the Fourier transform only has to be taken over any range of frequencies with length ω_s.

The summation in Eq. (2.44) is obviously for a discrete-signal of infinite length. In practical cases the duration of a measurement and hence the length of the signal will be finite. This case will be considered next.

Finite-time, discrete-time signals

Finally, for finite-time sampled signals we can limit the summation in Eq. (2.44) to the N samples that are available. Analogously to the continuous-time signals we can assume that the available samples represent one period with length $T_0 = NT$ of a periodic signal. Them, similar to Eq. (2.30) only non-zero Fourier coefficients U_l are found for specific frequencies ω_l. The outcome is

$$U_l = \sum_{k=0}^{N-1} u_k e^{-i\,\omega_l t_k}, \qquad u_k = \frac{1}{N}\sum_{l=0}^{N-1} U_l e^{i\,\omega_l t_k}. \tag{2.45}$$

As before, the signal u_k is defined at the discrete time instances

$$u_k = u(t_k) \quad \text{with} \quad t_k = kT, \tag{2.46}$$

and the Fourier transform is non-zero for (angular) frequencies

$$U_l = U(\omega_l) \quad \text{with} \quad \omega_l = \frac{l}{N}\omega_s = \frac{2\pi\, l}{NT} = \frac{2\pi\, l}{T_0}. \tag{2.47}$$

With these expressions Eq. (2.45) can be rewritten as

$$U_l = \sum_{k=0}^{N-1} u_k e^{-i\,2\pi\,kl/N}, \qquad u_k = \frac{1}{N}\sum_{l=0}^{N-1} U_l e^{i\,2\pi\,kl/N}, \tag{2.48}$$

which are purely mathematical relations between two finite series u_k and U_l without any references to physical quantities like time instances t_k and (angular) frequencies ω_l.

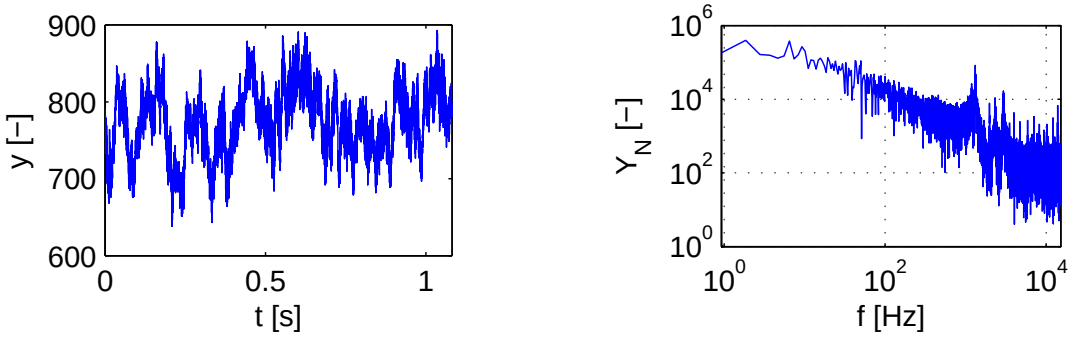
The sequence $\{U_l,\ l = 0...N-1\}$ is called the Discrete Fourier Transform (DFT) of the signal $\{u_k,\ k = 0...N-1\}$. This is the transform that is commonly available in many numerical software, like the `fft` and `ifft` commands in MATLAB. In case N is an integer power of 2 the very efficient Fast Fourier Transform (FFT) algorithm is available.

In general both series can included complex numbers. For the practical case that u_k represents a physical signal, it will consist of only real numbers. It can be shown that the DFT for such a real signal satisfies

$$U_l = U^*_{N-l}, \tag{2.49}$$

and U_0 (frequency 0 or DC) and $U_{N/2}$ (for frequency $\omega_s/2$, so for the Nyquist frequency ω_N) are real. Apparently, in the complex series U_l not all real and imaginary parts of the numbers are independent. More precisely, only N real numbers are significant in the series U_l, so one can conclude that the DFT transforms a real signal u_k of length N into a Fourier series U_l with also N significant real numbers. That also means that all necessary information about the Fourier transform is available in the coefficients $\{U_l,\ l = 0...N/2\}$.

Note that in MATLAB the first index for vectors and arrays equals 1. Hence the output of the `fft` command is a complex array of length N of which the elements with indices `1` to `N/2+1` contain the relevant data and belong to frequencies ranging from DC, $\omega_0 = 0$, to the Nyquist frequency $\omega_\mathrm{N} = \omega_s/2$.



(a) (b)

Figure 2.8: Position output signals from a piezo actuator driven mechanism: (a) Time domain, (b) Frequency domain.

To illustrate this, data from the piezo mechanism mentioned in chapter 1 is used (see also Appendix A.3). Figure 2.8(a) shows the recorded position output signal y_k as a function of the time $t_k = kT$. The sample period is $T = 33\,\mu$s and 32678 ($= 2^{15}$) samples have been recorded in a MATLAB variable `y`. The MATLAB command

```
yfft = fft(y);
```

will create a variable `yfft` with 32678 complex values. To show the frequency content as in Fig. 2.8(b), we need to:

- Consider only the lower part of the data in `yfft`, so ranging from `2:(N/2+1)` where `N` equals the total number of samples.
- Take the absolute value (`abs`).

- Use the correct scale for the frequencies on the horizontal axis. We can use either the angular frequency ω in rad/s of the frequency f in Hz. The angular frequency resolution $\Delta\omega$ follows from Eq. (2.47) and equals

$$\Delta\omega = \frac{1}{N}\omega_s = \frac{2\pi}{NT} = \frac{2\pi}{T_0}. \tag{2.50}$$

  Then the frequency resolution is

$$\Delta f = \frac{1}{N}f_s = \frac{1}{NT} = \frac{1}{T_0}, \tag{2.51}$$

  which is the inverse of the total measurement time $T_0 = NT$. The measurement time is $32678 \times 33\ \mu\text{s} = 1.08$ s, so the frequencies are 0.93 Hz apart. The maximum frequency is the Nyquist frequency, so half the sample frequency: $f_\text{N} = \frac{1}{2}f_s = 15$ kHz.

Finally, in the result shown in Fig. 2.8(b) logarithmic axes are used as is common for these frequency plots in order to show sufficient details in the full frequency range.

Spectral densities

The *energy spectral density function* $\Psi_u(\omega)$ of a finite energy signal continuous-time signal $u(t)$ is given by

$$\Psi_u(\omega) = |U(\omega)|^2, \tag{2.52}$$

where $U(\omega)$ is defined in Eq. (2.26). The total energy of the signal is then

$$\mathcal{E}_u = \int_{-\infty}^{\infty} u(t)^2 \mathrm{d}t = \frac{1}{2\pi}\int_{-\infty}^{\infty} \Psi_u(\omega)\mathrm{d}\omega. \tag{2.53}$$

The *power spectral density function* $\Phi_u(\omega)$ of a finite power signal continuous-time signal $u(t)$ is given by

$$\Phi_u(\omega) = \lim_{T_0\to\infty} \frac{1}{T_0}|U_{T_0}(\omega)|^2, \tag{2.54}$$

where $U_{T_0}(\omega)$ is defined in Eq. (2.29). The total power of the signal is then

$$\mathcal{P}_u = \lim_{T_0\to\infty} \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} u(t)^2 \mathrm{d}t = \frac{1}{2\pi}\int_{-\infty}^{\infty} \Phi_u(\omega)\mathrm{d}\omega. \tag{2.55}$$

For discrete-time signals these expressions read as follows. The energy spectral density function does not change

$$\Psi_u(\omega) = |U(\omega)|^2, \tag{2.56}$$

although the Fourier transform is now computed with Eq. (2.44) and the integral for the total energy of the signal in the frequency domain is limited to one period ω_s

$$\mathcal{E}_u = \sum_{k=-\infty}^{\infty} u_k^2 = \int_{2\pi/T} \Psi_u(\omega)\mathrm{d}\omega. \tag{2.57}$$

The power spectral density function is determined from the discrete Fourier transform in Eq. (2.45)

$$\Phi_u(\omega_l) = \lim_{N\to\infty} \frac{1}{N}|U_l|^2, \tag{2.58}$$

and the total power of the signal is then

$$\mathcal{P}_u = \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} u_k^2 = \int_{2\pi/T} \Phi_u(\omega) \mathrm{d}\omega. \tag{2.59}$$

For a signal u_k of finite length N the limit for $N \to \infty$ can not be evaluated practically. In e.g. MATLAB the power of a signal is computed from the available samples. The applied relations for the power are

$$\mathcal{P}_u = \sum_{k=0}^{N-1} u_k^2 = \sum_{l=0}^{N-1} \Phi_u(\omega_l), \tag{2.60}$$

in which the power spectral density function from Eq. (2.58) is used to compute the power in the frequency domain. Note that compared to Eq. (2.59) the scaling with the factor $1/N$ in the time domain is different.

A graph of $\Phi_u(\omega)$ is also denoted a *periodogram*. It is e.g. computed in the IDENT GUI with the option to show the "data spectra" of input and output signals.

Convolution

In Eq. (2.12) the convolution $y(t)$ of two continuous-time signals $g(t)$ and $u(t)$ has been defined as

$$y(t) = (g * u)(t) = \int_{-\infty}^{\infty} g(\tau) u(t-\tau) \mathrm{d}\tau. \tag{2.61}$$

The analogous expression for discrete-time signals is

$$y(k) = (g * u)(k) = \sum_{l=-\infty}^{\infty} g(l) u(k-l). \tag{2.62}$$

In either case taking the Fourier transforms leads to the simple relation

$$Y(\omega) = G(\omega) U(\omega), \tag{2.63}$$

indicating that convolution in the time domain leads to a multiplication in the frequency domain. An application of this property will be shown in Sect. 2.4.

A similar relation as Eq. (2.63) hold for the inverse transformation: Multiplication in the time domain is equivalent with convolution in the frequency domain. This property has been used to derive Eq. (2.39).

2.3.3 Stochastic signals

For stochastic systems producing stochastic signals we need to take into account that a signal like signal (c) in Fig. 2.6 is not just a function of time, but also depends on the realization of the stochastic process. We will consider so-called *stationary stochastic* processes. A signal $x(t)$ is such a process if

- at every time t the signal $x(t)$ is a random variable of which the value is determined by a time-invariant probability density function.
- the expectations of the signal $E(x(t))$ and the convolution $E(x(t)x(t-\tau))$ are independent of time t for every τ.

Furthermore, the signal $x(t)$ is called a *white noise* signal if

- x is a sequence of independent random variables, i.e. $E(x(t)x(t-\tau)) = 0$ for all $\tau \neq 0$.

Spectral densities are defined using the (auto)-covariance and cross-covariance functions defined as

$$R_x(\tau) = E(x(t) - \overline{x(t)})(x(t-\tau) - \overline{x(t-\tau)}), \tag{2.64}$$

and

$$R_{xy}(\tau) = E(x(t) - \overline{x(t)})(y(t-\tau) - \overline{y(t-\tau)}), \tag{2.65}$$

respectively, where

$$\overline{x(t)} = E(x(t)). \tag{2.66}$$

Then the power spectral density function or spectrum of $x(t)$ is

$$\Phi_x(\omega) = \int_{-\infty}^{\infty} R_x(\tau) e^{-i\omega\tau} \mathrm{d}\tau. \tag{2.67}$$

Also a power cross-spectral density function or cross-spectrum is defined according to

$$\Phi_{xy}(\omega) = \int_{-\infty}^{\infty} R_{xy}(\tau) e^{-i\omega\tau} \mathrm{d}\tau. \tag{2.68}$$

It is easy to show that for a signal with zero mean ($E(x(t)) = 0$)

$$\begin{array}{rcccl} R_x(0) & = & Ex(t)^2 & = & \dfrac{1}{2\pi}\displaystyle\int_{-\infty}^{\infty} \Phi_x(\omega)\mathrm{d}\omega \\ R_{xy}(0) & = & E(x(t)y(t)) & = & \dfrac{1}{2\pi}\displaystyle\int_{-\infty}^{\infty} \Phi_{xy}(\omega)\mathrm{d}\omega \end{array} \tag{2.69}$$

Discrete stochastic signals will be considered analogously. Of course, a stochastic signal x_k is defined only at the discrete time instances $t_k = kT$. Hence, integrals for the time t will be replaced by summations. The expressions for the cross-covariance functions do not change, except that the time variable t and τ are replaced by integers k and l, so

$$R_x(k) = E(x_l - \overline{x_l})(x_{l-k} - \overline{x_{l-k}}), \tag{2.70}$$

and

$$R_{xy}(k) = E(x_l - \overline{x_l})(y_{l-k} - \overline{y_{l-k}}), \tag{2.71}$$

respectively. Then the power spectral density function or spectrum of x_k is defined by means of a discrete-time Fourier transdform

$$\Phi_x(\omega) = \sum_{k=-\infty}^{\infty} R_x(k) e^{-i\omega kT}, \tag{2.72}$$

and in the integral of the inverse relation only an interval of length equal to the sample frequency has to be considered

$$R_x(k) = \frac{T}{2\pi} \int_{2\pi/T} \Phi_x(\omega) e^{i\omega kT} \mathrm{d}\omega. \tag{2.73}$$

2.4 Systems in time and frequency domain

A description of a system should specify how the output signal(s) y depend on the input signal(s) u. In this section two representations will be considered in the frequency and in the time domain respectively.

In section 2.3 the Fourier transform of signals has been outlined. It can be applied to obtain the Fourier transforms of input $U_N(\omega)$ and output $Y_N(\omega)$. The relation between these Fourier transforms gives a frequency domain representation of the system. Let us assume that a *Bode plot* is available for this system. It can be obtained e.g. from a transfer function of the system or it can be a *Frequency Response Function* (FRF) that has been measured experimentally as will be outlined later.

For a continuous time transfer function $G(s)$ the Bode plot is found by evaluating this complex function for $s = i\omega$ for all (positive) real angular frequencies ω. In a Bode plot two graphs show the amplitude $|G(i\omega)|$ and phase plot $\angle G(i\omega)$ as functions of ω, Fig. 2.2. As was outlined before (page 2-2), from these plots it follows immediately how the system behaves for a harmonic input. With this input and a stable system, the output will also be a harmonic signal after transient behaviour vanished. Furthermore the amplification equals the absolute value $|G(i\omega)|$ and the phase shift is found from the phase angle $\angle G(i\omega)$.

With this important property it should in principle be possible to compute the output of a LTI system: Transform the input into the frequency domain, multiply with the Bode plot or FRF, transform the result back to the time domain. Note that in the case of sampled data, it is more appropriate to consider the discrete transfer function $G(z)$ that is evaluated at the unit circle, so $G(e^{i\omega T})$.

A time domain representation is given by the (unit) *impulse* response of a system. This impulse response is the output of the system when a unit impulse is applied on the input. For a discrete time signal, the impulse is defined as a signal that is 1 at only time step $t = 0$ and 0 for all other time steps. For a continuous time signal, the Dirac delta (δ) function provides a mathematical description of a very short input signal of unit magnitude. In either case the impulse response is specified as an (infinite) number of samples g_k or as a function $g(t)$ respectively. Note that *causal* systems can not predict the future and then the impulse response only has to be considered for $k \geq 0$ or $t \geq 0$.

An arbitrary continuous time signal $u(t)$ can be considered as an infinite number of impulses of some magnitude. Each response will result in an impulse response in the output. For an LTI system all these responses can be combined as in the convolution integral of Eq. (2.12) on page 2-3. Analogously, a discrete time signal u_k is also a series of impulses, so also in this case the output can be found from the convolution summation Eq. (2.62). Obviously, the impulse response provides another way to represent the behaviour of an LTI system.

This short overview introduced two representations for a system, namely the impulse response in the time domain and the Bode plot or FRF in the frequency domain. A few remarks have to be added:

- The impulse response and the Bode plot are related via Eq. (2.63), which is true for any convolution. It essentially means that the the Bode plot of a system can be computed directly from the impulse response using a Fourier transform and vice versa.

- It is, in principle, possible to compute the output of a system from the input and a representation either as an impulse response or as a Bode plot. The reason to point this out is more for theoretical than for practical reasons. It shows nevertheless that in either representation, in principle, all properties of the system are included. In the next chapter it will be discussed how impulse responses and Bode plots can be determined from experimental data.

- Note that these representations are not very “efficient”. For typical systems, both consist of a fairly large set of numbers. So in general, it is not very efficient to compute the output signal with such a representations. The impulse response and Bode plots will be referred to as *non-parametric models* of a system. The so-called *parametric models* that will be discussed in subsequent chapters

provide a more "compact" representation of the system and are better suited for output simulations.

2.5 Systems and models

With system identification we look for models of a system. A typical configuration is shown in Fig. 2.9. The system G_0 relates its output to the input $u(k)$. In general, this "true" output can not be measured.

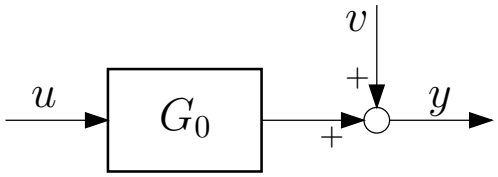


Figure 2.9: Typical system G_0 with input u and output y disturbed by noise v.

In stead we assume that we know the input $u(k)$ and the output $y(k)$ which includes an unknown noise signal $v(k)$. An important question will be whether we can obtain a "good" model $\hat{G}$ of the system from N measured samples of input and output. To define what is meant with a "good" model we assume that G depends on a vector θ with n real parameters and the "true" system is described by the (unknown) vector θ_0. We determine an estimation θ_N which is

- unbiased if $E\theta_N = \theta_0$.
- consistent if the probability distribution of θ_N approaches a δ-function for $N \to \infty$.

3 Non-parametric system identification

3.1 Introduction

As a first example of identified models the results of so-called non-parametric system identification will be discussed. These models are expressed either in the time domain as impulse or step responses, or in the frequency domain leading to spectral models e.g. in the format of Bode diagrams (or FRF). These techniques are denoted "system identification" as the obtained models are primarily mathematical and no physical parameters are obtained. The term "non-parametric" may be misleading as also a non-parametric model is composed of a list of parameters. E.g. the impulse response g_k gives the response of a system at a series of time steps t_k. These g_k are essentially the parameters of the system to be identified. For a real system, quite a number of parameters will have to be taken into account. In contrast, it will be outlined later that parametric identification yields compact models with less parameters. Those parametric models are much better suited for e.g. simulations. In principle an impulse response can be used to simulate the output of a linear time-invariant system for any input, but a transfer function is much easier to use and gives also more insight e.g. in the damping of an eigenfrequency or for the design of a controller for a closed loop system.

For the latter reasons these parametric models may seem "better". Indeed (system) identification mostly doesn't stop with non-parametric models. Nevertheless there are several motivations to include non-parametric models as one of the identified models. Consistency of parametric models will appear to be far from trivial and therefore parametric models may give quite misleading results. The non-parametric models, so impulse response and/or spectral model, can give insight about specific dynamic behaviour of a system like a resonance frequency. Such insight can be of help to detect erroneous results in a parametric identification. For that reason, non-parametric identification is often the fist step is the whole identification procedure.

In this chapter some of the mathematical theory will be outlined and illustrated with examples. Tools from MATLAB's `ident` toolbox will be used for the identification. Two techniques are available for the spectral models in the frequency domain: `spa` and `etfe`. The `cra` commands can be used to identify an impulse response. Note that the toolbox also provides the `impulse` and `step` command for the time domain analysis, but these commands use a different algorithm.

Before the identification will be discussed in detail, it is important to consider the system that is depicted in figure 3.1. It represents a system G_0 with known input u. An unknown disturbance v is added to the output of G_0 to obtain a measurable output signal y. The description of the system G_0 somehow has to account for the way the input u affects the (measurable) output y. Then the remaining part in y, which equals the disturbance v, has no relation with this input u.

At first sight that may seam to be a quite reasonable interpretation of a real system. For the analysis to follow some extra assumptions are made that will make this less obvious. In particular, it is assumed that the system is linear and can e.g. be written as a discrete time transfer function $G_0(z)$. Then

$$y(t) = G_0(z)\, u(t) + v(t). \tag{3.1}$$

Looking at a real system this means that some of the measured output y is the results of the linear

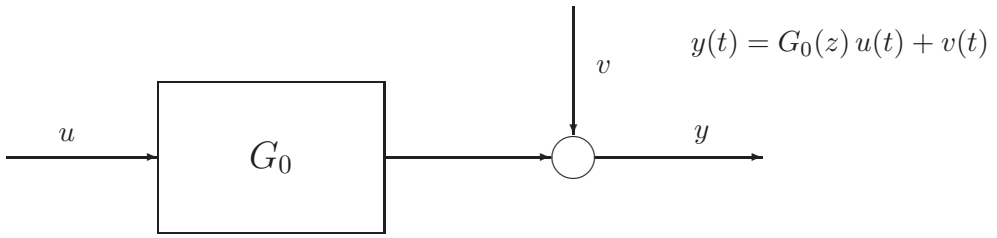


Figure 3.1: Schematic representation of a system.

system $G_0(z)$ with input u. All the rest is attributed to the (unknown) "noise" signal v. This includes measurement noise and the output from any non-measured inputs. That is fine as long as it is reasonable that such sources are indeed independent from the (measured) input u. However, *all* contributions to the output y that are not generated by the linear term $G_0(z)\,u(t)$ need to be part of v and this may include unmodelled and/or non-linear system behaviour. In that case there is some relation between v and u, so it is no longer correct to assume that u and v are not related to each other.

3.2 Correlation analysis

Assuming that the representation in figure 3.1 with the linear relation (3.1) are valid, then the goal of the identification is to determine the linear system $G_0(z)$. One way to describe the system is by using the *Impulse Response* $g_0(k), k = 0, 1, 2, ...$ as the output y can be written as

$$y(t) = \sum_{k=0}^{\infty} g_0(k)u(t-k) + v(t) \qquad (t = 0, 1, 2, ...). \tag{3.2}$$

By comparing Eqs. (3.1) and (3.2) the linear system $G_0(z)$ can be written as

$$G_0(z) = \sum_{k=0}^{\infty} g_0(k)z^{-k}. \tag{3.3}$$

Note that a drawback of this expression is that an impulse response of *infinite* length is used. For an identification the input u and output y are recorded and the system $G_0(z)$ will be modelled with a *Finite Impulse Response* (FIR) $\hat{g}(k)$, $k = 0, 1, 2, ..., M$. In this notation a hat $\hat{(\cdot)}$ is used to indicate that this response is an estimate for the system computed from the measured data. For a sufficiently high order M it is then assumed that the output can be approximated with

$$y(t) \approx \sum_{k=0}^{M} \hat{g}(k)u(t-k) \qquad (t = 0, 1, 2, ...). \tag{3.4}$$

For a causal system the lower limit of the summation can not be less than zero. Nevertheless, during system identification the lower limit of the summation is often taken less than 0 (e.g. $-m$) to verify that there is indeed only a causal relation between $u(t)$ and $y(t)$. Any (unexpected) non-causal behaviour can e.g. originate from experimental errors or the presence of feedback. Experimental errors may occur when both input and output have to be measured and there is a longer delay in the measurement of the input than in the measurement of the output. Then y may appear to be "ahead" of u. The same observation can arise from feedback. If the input u is somehow computed from the output y then it may seem as if there is non-causal behaviour.

Next the estimator $\hat{g}(k)$ in Eq. (3.4) has to be computed from the input and output data u and y respectively. Correlation analysis will be applied for this purpose. The expression is multiplied with

$u(t-\tau)$ and the expectation is computed. It will be assumed that $u(t)$ en $v(t)$ are uncorrelated, so $R_{vu}(\tau) = 0$. Then the so-called Wiener-Hopf equation is obtained:

$$R_{yu}(\tau) = \sum_{k=0}^{\infty} g_0(k) R_u(\tau - k). \tag{3.5}$$

Note that the assumption $R_{vu}(\tau) = 0$ is essential for the applicability of this equation. This assumption is however violated when the disturbance v includes non-linear behaviour of the system that is not in the linear system $G_0(z)$. It is also violated if there is feedback from y to u.

To solve $\hat{g}(k)$ from Eq. (3.5) two cases will be distinguished. The easy case is when $u(t)$ is a white noise signal. Then the autocovariance equals

$$R_u(\tau) = \sigma_u^2 \delta(\tau), \tag{3.6}$$

where σ_u^2 is the variance of the noise signal $u(t)$. It follows immediately that

$$\hat{g}(\tau) = \frac{\hat{R}_{yu}(\tau)}{\sigma_u^2}, \tag{3.7}$$

in which $\hat{R}_{yu}(\tau)$ is an estimator for the cross-covariance. This estimator can be computed from N measurements with the *sample covariance function*

$$\hat{R}_{yu}^N(\tau) = \frac{1}{N} \sum_{t=\tau}^{N} y(t) u(t-\tau). \tag{3.8}$$

The summation in this expression has $N - |\tau|$ terms. The division by N introduces a bias, but still the estimator is asymptotically unbiased, so for $N \to \infty$.

In the case $u(t)$ is *not* a white noise signal some more effort is needed to solve the Wiener-Hopf equation (3.5). One way is to estimate the autocovariance e.g. with

$$\hat{R}_u^N(\tau) = \frac{1}{N} \sum_{t=\tau}^{N} u(t) u(t-\tau) \tag{3.9}$$

and next solve the linear set of M equations for $\hat{g}(k)$

$$\hat{R}_{yu}^N(\tau) = \sum_{k=1}^{M} \hat{g}(k) \hat{R}_u^N(\tau - k) \tag{3.10}$$

Although this solution may seem quite straightforward, it involves many computations for the estimates of the covariances and finally to solve the equations.

An alternative approach is to try to modify the data such that the relatively easy solution of Eq. (3.7) can be applied. A requirement is a white input signal. This can be accomplished by a procedure that will be applied more often, namely filtering both input and output data with some linear filter $L(z)$ to obtain

$$u_F(t) = L(z)u(t) \quad \text{and} \quad y_F(t) = L(z)y(t). \tag{3.11}$$

The filtered output can be written as

$$y_F(t) = L(z) \left(G_0(z)u(t) + v(t) \right). \tag{3.12}$$

For a linear system then also

$$y_F(t) = G_0(z)u_F(t) + L(z)v(t), \tag{3.13}$$

so the relation between the filtered input $u_F(t)$ and output $y_F(t)$ is given by the same system $G_0(z)$ and a disturbance signal that equals $L(z)v(t)$. Apparently, the signals $u_F(t)$ and $y_F(t)$ can also be used to

estimate the impulse response $\hat{g}(k)$. In order to apply Eq. (3.7) the filter $L(z)$ has to be chosen such that the filtered input signal $u_F(t)$ is a white noise signal.

The exact procedure to determine such a *pre-whitening* filter $L(z)$ will be outlined later. For the moment it is just indicated that in the `ident` toolbox this is accomplished by using a linear model of order n (default in ident $n = 10$):

$$L(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + ... + a_n z^{-n}. \tag{3.14}$$

The n parameters $a_1, a_2, ..., a_n$ are found by minimising the sum

$$\frac{1}{N} \sum_{k=1}^{N} u_F^2(k). \tag{3.15}$$

This is an example of a so-called *linear least squares* fit that will be used frequently in this course and will be dealt with in more detail later.

3.2.1 Example: Piezo mechanism

As a first example the experimental data from a piezo mechanism (appendix A.3) is analysed. One important aspect was not explicitly outlined so far, but all the equations for linear systems derived from Eq. (3.1) do not include constant offsets in signals. That means that a stationary zero input will result in a zero output. In simulations that is easy to accomplish, but in experimental data there may be all kind of sources for constant or even slowly varying offsets.

In the raw data of the piezo mechanism the absolute output position y has no well defined meaning. At some position the sensor will indeed give a zero output, but for the considered experiment it was only checked that the position of the mechanism was within the measurement range of the sensor and the zero has no special meaning. Consequently, the mean of the output signal has an arbitrary non-zero mean as is clear from e.g. Fig. A.1. The input current of the piezo actuator u can be positive and negative. The measurement software with which the data were collected, only uses unsigned integer values ranging from 0 to 4095. Zero current is associated with a integer value at or near 2048.

Obviously, *pre-processing* of the data is necessary prior to identification. This pre-processing will be explained in more detail later. For the estimation of the impulse response it is only necessary to remove the non-zero mean in both input and output data. This can be easily carried out with the `detrend` command of MATLAB's `ident` toolbox:

```
load piezo;                % Load data
Ts = 33e-6;                % Sample time
u=piezo(:,2);              % Input
y=piezo(:,1);              % Output
piezo = iddata(y,u,Ts);    % Create iddata DATA OBJECT
piezod = detrend(piezo);   % Remove means in input and output
```

Figure 3.2 shows a `plot` of the data set after the `detrend` operation.

Now the data is ready for the correlation analysis. The `cra` command can be used to automate the procedure. Typing

```
[ir,R,cl]=cra(piezod,200,10,2);
```

gives the plots as shown in Fig. 3.3. The last parameter (2) of the `cra` command indicates that all these four graphs should be drawn. The second parameter of the command specifies that 200 samples of the impulse response should be computed. According to the third parameter a 10^{th} order pre-whitening filter should be used. It may be questioned whether this is necessary as the PRBS input signal u resembles already white noise.

The upper right graph in Fig. 3.3 gives the autocovariance $R_u(\tau)$ after application of the pre-whitening filter. Irrespective of the whiteness of the original signal, it is clear from this figure that

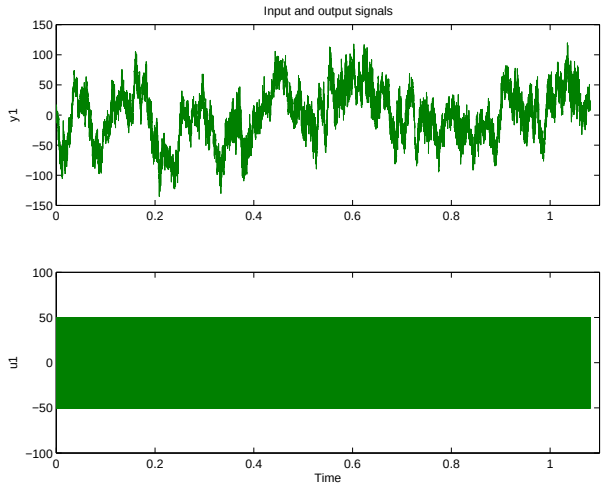


Figure 3.2: Input and output signal of the piezo system after removing the means.

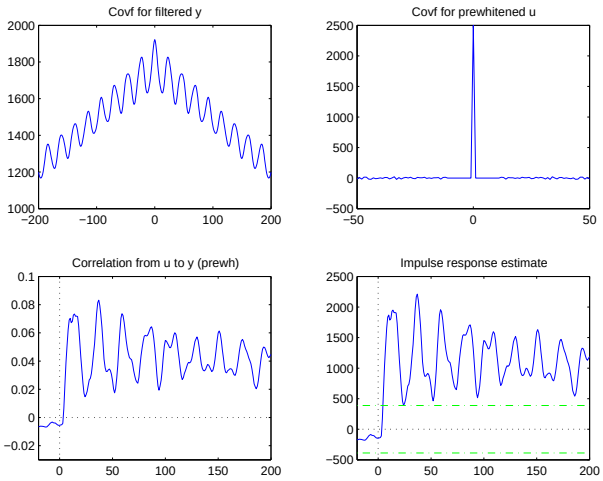


Figure 3.3: Result of the `cra` analysis of the piezo system (after some small changes of the axes limits).

the filtered input is white as there is a dominant peak for $\tau = 0$ and for other τ only a small noise remains. The upper left graph is the autocovariance of the (filtered) output $R_y(\tau)$ and is not used to estimate the impulse response of the system. For that estimate the cross-covariance in the lower left graph is more important. After scaling it gives the estimated impulse response in the lower right graph. For clarity, the $\tau = 0$ axis is indicated with a dotted line. It appears that the impulse response exhibits only causal behaviour as the estimate is close to zero for $\tau < 0$. This is an important check as non-causal behaviour may be caused by measurement or analysis errors.

Note that all horizontal axes in Fig. 3.3 count the time samples. In order to obtain a real time scale these (integer) values must be multiplied with the sample time $T = 33$ μs.

3.2.2 Example: Known system

The advantage of analysing a known system is that the estimate can be compared with the known properties of the system. This is demonstrated for the second order ARMAX system G_0 of appendix A.1. As in the appendix data is simulated for $N = 4096$ time samples. The known impulse response and the results according to `cra` analyses are given in Fig. 3.4. The two solid lines have many common features indicating that the `cra` analysis with a 10^{th} order pre-whitening filter agrees well with the "real" system G_0. The noise in the estimate is visible for the response after 15 s.

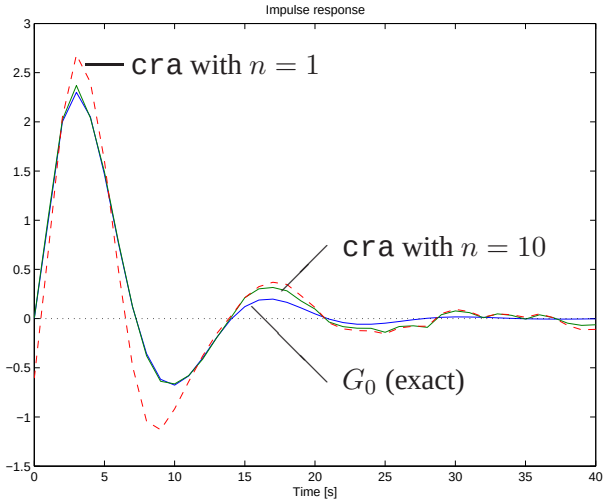


Figure 3.4: Impulse response of a known system and result of `cra` analyses.

For this data it is important to apply a pre-whitening of sufficient order. This is illustrated by the dashed line that results from a `cra` analysis with only a first order pre-whitening filter. The input u is somehow limited to a frequency up to 0.3 times the sample frequency and therefore is not a white noise signal. A first order pre-whitening filter is too simple to assure that the filtered input u_F becomes white. As `cra` applies Eq. (3.7) irrespective of the whiteness of the input signal, an erroneous result is obtained.

3.3 Spectral analysis

The goal of non-parametric identification in the frequency domain is to obtain the frequency response of a system, so the Bode plot. It can be measured directly by exciting a system with a harmonic signal and measuring the output (after transients are damped). That measurement gives immediately the frequency response. For each frequency of the input signal the gain and phase shift are determined by comparing input and output. Thus for each frequency a complex number is found of which the absolute value equals the gain and the phase corresponds to the measured phase shift.

In this section the frequency domain representation is estimated from time domain data. The system of Fig. 3.1 is considered once more and Eq. (3.1) is again the starting point. When we ignore the presence of the disturbance $v(t)$, the Fourier transform of Eq. (3.1) gives

$$Y(\omega) = G_0(e^{i\omega T})U(\omega). \tag{3.16}$$

Then the gain and phase of the system follow immediately from

$$G_0(e^{i\omega T}) = \frac{Y(\omega)}{U(\omega)}. \tag{3.17}$$

For a limited number of measurements N, the Fourier transforms of input and output are computed from those measurements and an estimator for the frequency response is

$$\hat{G}_N(e^{i\omega T}) = \frac{Y_N(\omega)}{U_N(\omega)}. \tag{3.18}$$

However, in a real measurement the disturbance always has to be taken into account. Then the estimator is not the exact frequency response, but instead

$$\hat{G}_N(e^{i\omega T}) = G_0(e^{i\omega T}) + \frac{V_N(\omega)}{U_N(\omega)} \tag{3.19}$$

is found. From this equation it can be seen that the estimator $\hat{G}_N$ [6, Chapter 6]

(a) is unbiased. The expectation of the effect of $v(t)$ is zero, so the expectation of $\hat{G}_N$ is equal to the "real" value.

(b) has an asymptotic variance

$$\Phi_v(\omega)/\frac{1}{N}|U_N(\omega)|^2$$

which does not decrease to zero for large N.

(c) is asymptotically uncorrelated for different frequencies ω.

That means that although the estimator $\hat{G}_N$ is unbiased, it is not consistent as the variance does not decrease. At first sight that may be surprising as apparently increasing the number of data points N does not lead to a less noisy estimate. An explanation for this behaviour is that the Fourier transform of more data will be used to estimate the frequency response at more (=$N/2$) frequencies. And according to property (c) of the estimator $\hat{G}_N$ neighbouring frequency "channels" are not correlated, so in each channel the noise is not decreased.

There are roughly two possible solutions to obtain a "better" estimate:

1. Define a periodic excitation with a fixed period N_0 and consider an increasing number of measurements $N = rN_0$ by increasing r. Carry out the spectral analysis for each period and compute the average to obtain a "good" estimator in $N_0/2$ frequencies. This estimator is consistent.

2. Smoothen the spectrum in the frequency domain. Effectively, the smoothing operation somehow combines the uncorrelated neighbouring frequencies. As a result the variance is reduced.

Example: Piezo mechanism

To illustrate the variance of the Fourier transform, the frequency content of the output (position) signal of the piezo mechanism is considered. The lower curve in Fig. 3.5 shows the absolute value of Fourier transform of this signal without any further processing. The frequency content is computed in 16384 frequencies and obviously the result is quite noisy.

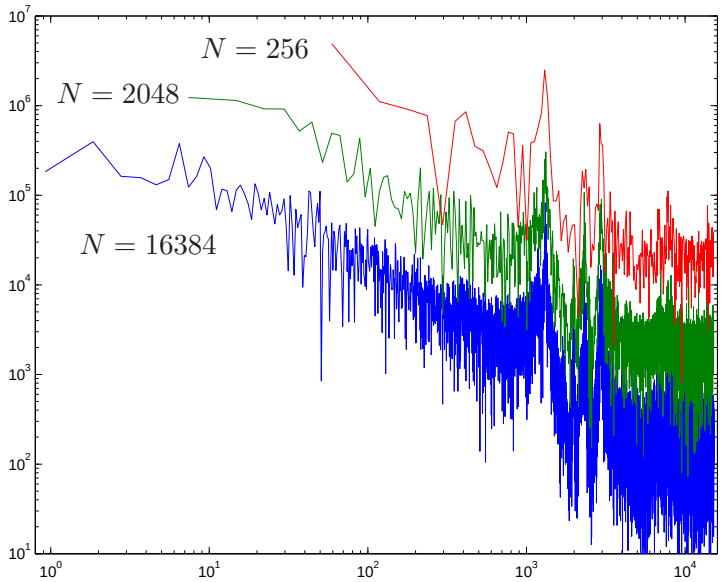


Figure 3.5: Frequency content of the output (position) signal of the piezo mechanism estimated with Fourier transforms. The number of frequencies N is given for each graph. The upper two graphs are shifted vertically.

Next the frequency content is computed for less frequencies. The data is split in a number of parts of equal length. For each part the Fourier transform is computed and all results are averaged. The maximum frequency in the spectrum does not change this way and remains equal to the Nyquist frequency. The frequency resolution of course decreases as the duration of the measurements for each part is less than for all data. Fig. 3.5 includes the results of this analysis for 2048 frequencies (averaging Fourier transforms from 8 parts) and for 256 frequencies (averaging Fourier transforms from 64 parts). For the latter result still large fluctuations remain, but overall it can be concluded that the variance is indeed reduced.

Figure 3.6 shows an example with results of the second approach. The lower curve once more is the unprocessed Fourier transform for all 16384 frequencies. Next smoothing is applied with a so-called Hamming filter. This filter combines m neighbouring frequencies by applying weights w_j that are defined with a cosine function

$$w_j = 0.54 - 0.46 \cos \frac{2\pi j}{m-1} \quad \text{for } j = 0, 1, 2, ..., m-1, \tag{3.20}$$

where j indicates the index of the frequency and m determines the width of the *window* in which the averaging takes place [10]. For each frequency ω_l for which the estimate $\hat{G}_N(e^{i\omega_l T})$ is defined, a *filtered* estimate $\hat{G}^f_N(e^{i\omega_l T})$ is computed. With MATLAB's `filtfilt` command this filtering is carried out such that peaks in the spectrum do not move to other frequencies. First the spectrum is filtered in forward direction and an intermediate estimate $\hat{G}^i_N(e^{i\omega_l T})$ is computed with

$$\hat{G}^i_N(e^{i\omega_l T}) = \sum_{j=0}^{m-1} w_j \hat{G}_N(e^{i\omega_{l+j} T}) / \sum_{j=0}^{m-1} w_j. \tag{3.21}$$

This result is processed with a similar procedure in the reverse direction

$$\hat{G}^f_N(e^{i\omega_l T}) = \sum_{j=0}^{m-1} w_j \hat{G}^i_N(e^{i\omega_{l-j} T}) / \sum_{j=0}^{m-1} w_j, \tag{3.22}$$

to compute the filtered estimate $\hat{G}^f_N(e^{i\omega_l T})$ [10].

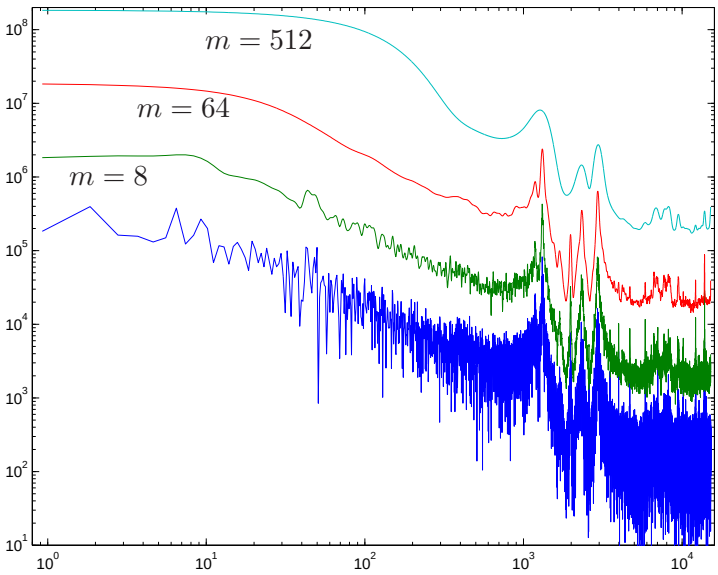


Figure 3.6: Frequency content of the output (position) signal of the piezo mechanism estimated with Fourier transforms. The upper three graphs are filtered with a Hamming filter of which the width is indicated. These graphs are shifted vertically.

For the three filtered curves in Fig. 3.6 windows with increasing widths of 8, 64 and 512 are used. Obviously a larger window results in a smoother estimate of the frequency content. However, too much

smoothing will broaden peaks in the spectrum and hide features in the data as in the smoothest curve in Fig. 3.6.

With the `ident` toolbox "data spectra" can be easily computed from the GUI. By default the so-called *periodograms* of the data are shown, i.e. the absolute square of the Fourier transforms of the data. It is possible to select the frequency scale and to apply filtering. Figure 3.7 shows the default periodograms for the input and output data of the piezo mechanism. The spectrum of the input signal is

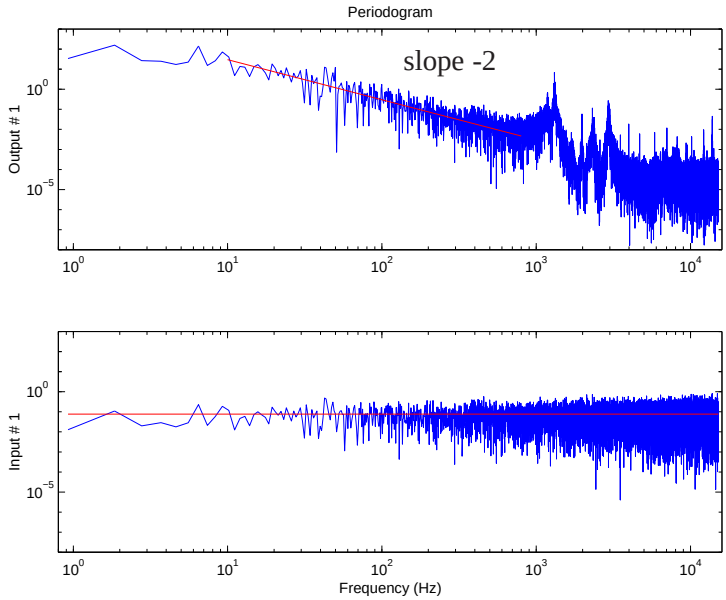


Figure 3.7: Data spectra or periodogram computed with `ident` for the input and output data of the piezo mechanism.

not exactly constant, but is still "reasonable" white as the excitation of the system is more or less the same for any frequency band between the lowest frequency and the Nyquist frequency. As the input signal is reasonably flat, the output signal should resemble the magnitude of the Bode plot for this system. It is expected that at low frequencies the integration of the input current gives rise to a pure integrator in the transfer function (see appendix A.3). This integrator will show up as a part with slope -1 in the magnitude Bode plot. In the periodogram the absolute square of the Fourier transform is plotted, so a slope -2 is expected. In Fig. 3.7 this slope is indeed visible in the output data.

Spectral analysis in ident

In the `ident` toolbox two commands are provided for the spectral analysis: `etfe` and `spa`.

The `etfe` command computes the Empirical Transfer Function Estimate, which is essentially the procedure outlined above. The frequency response of system G is estimated with Fourier transforms of input and output data

$$\hat{G}_N(e^{i\omega T}) = \frac{Y_N(\omega)}{U_N(\omega)}. \tag{3.23}$$

For this expression to hold, the only assumption imposed on the system is linearity of the system. In the case smoothing is applied, the nominator and denominator in this expression are first multiplied with the complex conjugate of $U_N(\omega)$, so

$$\hat{G}_N(e^{i\omega T}) = \frac{Y_N(\omega)U_N^*(\omega)}{|U_N(\omega)|^2}. \tag{3.24}$$

Both nominator and denominator are filtered with a Hamming filter before the ratio is computed. The level of smoothing depends on a parameter `M` that can be specified as the second parameter of the `etfe` command. The width of Hamming window equals the number of data points N divided by the parameter `M` plus one. So a small `M` means more smoothing. Finally only a limited number (default: 128) of frequencies is presented [7].

The `spa` command performs the SPectral Analysis with a somewhat different approach. Assuming the system can be describes by Eq. (3.1), covariances are used to estimate the frequency functions and spectrum. If input u and disturbance v are independent, then [7]

$$\Phi_y(\omega) = |G(e^{i\omega T})|^2 \Phi_u(\omega) + \Phi_v(\omega), \tag{3.25}$$

and

$$\Phi_{yu}(\omega) = G(e^{i\omega T}) \Phi_u(\omega), \tag{3.26}$$

where the $\Phi_{yu}(\omega)$, $\Phi_y(\omega)$, $\Phi_u(\omega)$ and $\Phi_v(\omega)$ are the spectra of the corresponding cross-covariance $R_{yu}(\tau)$ and autocovariances $R_y(\tau)$, $R_u(\tau)$ and $R_v(\tau)$ respectively. So the frequency response of system G can be estimated from estimates of Fourier transforms of the covariances

$$\hat{G}_N(e^{i\omega T}) = \frac{\hat{\Phi}_{yu}(\omega)}{\hat{\Phi}_u(\omega)} \tag{3.27}$$

and the disturbance spectrum from

$$\hat{\Phi}_v(\omega) = \hat{\Phi}_y(\omega) - \frac{|\hat{\Phi}_{yu}(\omega)|^2}{\hat{\Phi}_u(\omega)}. \tag{3.28}$$

This expression can also be written as

$$\hat{\Phi}_v(\omega) = \hat{\Phi}_y(\omega) \left[1 - (\hat{\kappa}_{yu}(\omega))^2\right], \tag{3.29}$$

in which

$$\hat{\kappa}_{yu}(\omega) = \sqrt{\frac{|\hat{\Phi}_{yu}(\omega)|^2}{\hat{\Phi}_y(\omega)\hat{\Phi}_u(\omega)}} \tag{3.30}$$

is called the *coherence spectrum* between output y and input u. It is a function of the frequency that is in the range from 0 to 1 and it gives a measure for the correlation coefficient between input and output. The coefficient equals 1 when there is a perfect (i.e. noise free) correlation between input and output, whereas 0 indicated that the output is completely dominated by the disturbance v.

What remains is the computation of the estimates for the Φ. These spectra can be estimated from estimates of the covariances. As before the sample covariance function can be used, which equals

$$\hat{R}^N_{yu}(\tau) = \frac{1}{N} \sum_{t=\tau}^{N} y(t)u(t-\tau). \tag{3.31}$$

for the cross-covariance and analog expressions for the autocovariances.

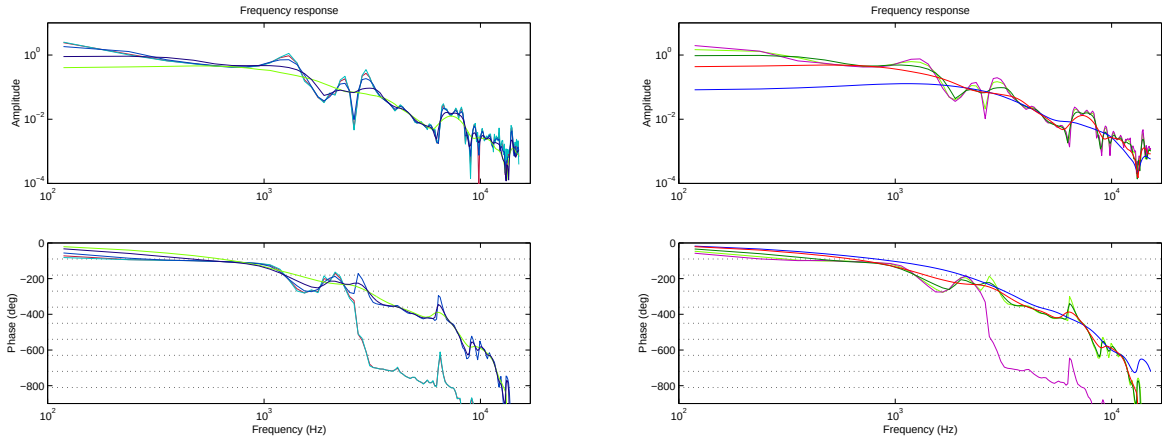
The `spa` command computes the estimates only in a limited number (default: 128) of frequencies. The level of smoothing again depends on a parameter `M` (default: 30). This parameters specifies how many terms are considered in the covariances. Before the Fourier transforms are computed the covariances are filtered with a Hamming filter of which the width matches the length of the covariances (so from $-$`M` to `M`). This way a similar behaviour as for the `etfe` command is obtained: Small `M` means more smoothing.

Comparing both methods, it is not always straightforward to predict which method will perform best and it is often not a large burden to try both. In either case the desired level of smoothing can be chosen.

The methods differ mainly in the exact way the smoothing is performed and which method is the “best” depends on the data and system at hand.

In some cases a preference can be expected. For systems with clear peaks in the spectrum often less smoothing should be applied to avoid artificial broadening of the peaks. Then `etfe` is preferred as it is more efficient for large values of the smoothing parameter `M`. On the other hand `spa` estimates also the noise spectrum with Eq. (3.28), which is an advantage if this noise spectrum is of interest.

Example: Piezo mechanism



(a) ETFE with M = 15, 30, 60, 90, 120 (b) SPA with M = 15, 30, 60, 90, 120

Figure 3.8: Spectral analyses of the piezo mechanism with `ident`.

Figure 3.8 shows the results of spectral analyses of the data from the piezo mechanism. Both `etfe` and `spa` analyses are shown, each with five values for the smoothing parameter `M`. For small values of `M` there is clearly too much smoothing as all features in the spectra are hidden. For the largest values of `M` the noise level becomes apparent. For this data it is difficult to determine the “real” width of the resonance peak near 2 kHz. One of the largest `M` values probably gives the best result, but one has to be aware that even in those spectra the peak may be broadened.

Note also the differences between the phases of some of the curves. The phase is computed as the phase angle of a complex number, which is of course not uniquely defined and can be different with multiples of 360°.

Example: Known system

The known second order system of appendix A.1 can be used to analogously to show the outcome of the spectral analyses in `ident`. That procedure is quite straightforward and is left as an exercise for the reader.

Instead, this system is used to once more show the behaviour of the variances of the Fourier transforms and the means to reduce them. This can be seen clearly in the `etfe` estimates in Figs. 3.9 and 3.10. In both figures the left graphs show the estimated frequency response without any processing of the data. Equation (3.23) is applied in which the Fourier transforms in all $N/2$ frequencies are evaluated and the ratio between output and input is the estimated frequency response. The solid line is the known $G_0(e^{i\omega T})$. Every dot represents the estimated frequency response for one frequency. Clearly the variances in the Fourier transforms result in errors in the estimate. Both suggestions that where given on page 3-7 for the reduction of the variances in the Fourier transform estimates will be discussed next.

First the number of frequencies in the Fourier transform is limited while the number of data points N remains the same. The procedure is as follows: Only the first N/r samples of the original input signal are used. These samples are repeated r times. Then the Fourier transform of the input signal is

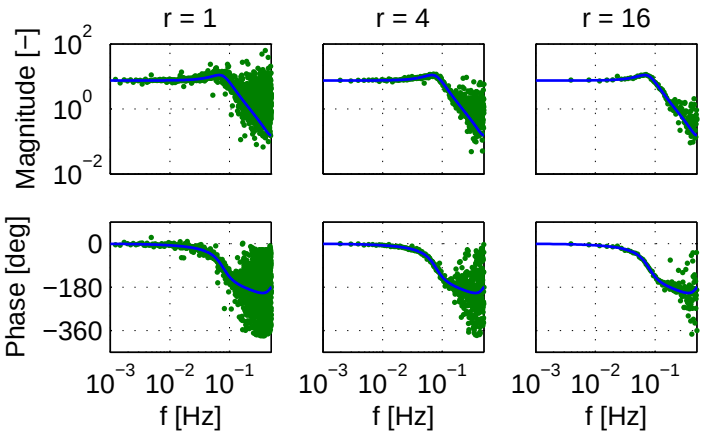


Figure 3.9: Spectral analyses of the known system. ETFE results are shown using periodic input data.

only non-zero in $N/2r$ frequencies. Next only in these frequencies the frequency response is estimated. Figure 3.9 shows the result for two values of r larger than 1. It can be seen clearly that the number of dots, so the number of frequencies with an estimate, is reduced. Furthermore the variance of the points with respect the exact $G_0(e^{i\omega T})$ also decreases.

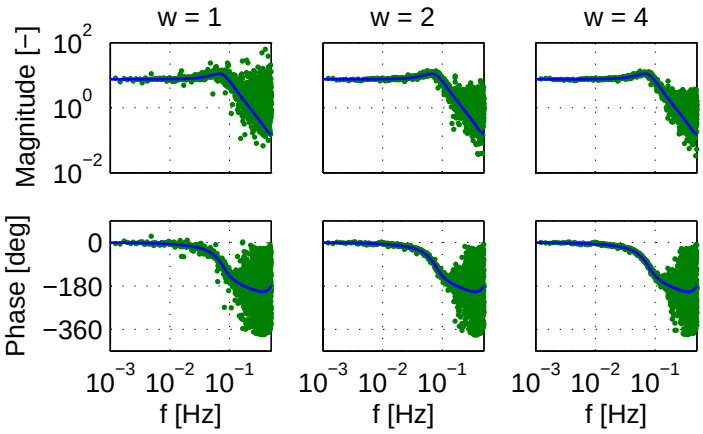


Figure 3.10: Spectral analyses of the known system. ETFE results are shown with filtering.

Alternatively, the original frequency response estimate can be filtered. Figure 3.10 shows the result in which two Hamming filters of increasing width w are used. The number of dots is not decreased, but clearly the variance is reduced considerably after filtering.

3.4 Closure

The outcome of the non-parametric identification outlined in this chapter is either an impulse response in the time domain or a frequency response in the frequency domain. It was pointed out in the introduction that these representations can be seen as models with a fairly large set of parameters. There are several ways to obtain models with a small number of parameters. In a later chapter the parametric identification will be discussed. Compact models are then immediately fitted on the experimental data. An alternative approach is to try to recognise "features" in the non-parametric models obtained so far.

As for the spectral models one can try to recognise known features in the frequency response. A known example is the characteristic Bode plot of a second order system. A resonance peak reveals the eigenfrequency and the height and width of the peak are associated with the damping. More elaborate

techniques are available e.g. to fit arbitrary transfer functions with a frequency response. This frequency response can either be measured directly or can be the result of the spectral analysis. This so-called frequency domain (system) identification will also be addressed later in this course.

Similarly, one can try to recognise features in the measured or identified impulse (or step) response of the system. Again second order systems are known to show characteristic behaviour. A more general approach is provided by the so-called realisation algorithms to be discussed later.

4 Linear regression and least squares estimate

4.1 Linear regression

Before we will try to estimate dynamic models with a "small number of parameters", we will first address the more general topic of linear regression. To start with, the considered models do not include any dynamic behaviour. It will appear that the extension to dynamic models is straightforward.

4.1.1 Regression

We consider the system depicted in Fig. 4.1. It is modelled by means of a function $g(\boldsymbol{\varphi})$ that depends on a number of independent variables, the *regressors*, $\varphi_1, \varphi_2, ..., \varphi_d$. This function is parameterised with a set of parameters collected in $\boldsymbol{\theta}$. As illustrated in the figure, the regressors in $\boldsymbol{\varphi}$ may in turn depend on one or more input quantities x.

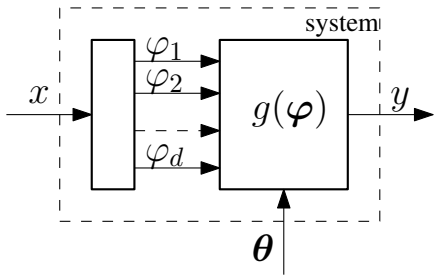


Figure 4.1: A system which the output y depends on an input x.

The goal of *regression* is to find an expression for the function $g(\boldsymbol{\varphi})$ such that it can be used to *predict* the output y. For that purpose, the output y is measured as well as the regressors

$$\boldsymbol{\varphi} = \begin{bmatrix} \varphi_1 \\ \vdots \\ \varphi_d \end{bmatrix}. \tag{4.1}$$

Note that these regressors may be measured directly or they can be computed with known relations from one or more measured inputs x as in Fig. 4.1. In any case, we would like to find the function of the regressors $g(\boldsymbol{\varphi})$ that minimises the difference $y - g(\boldsymbol{\varphi})$ in some sense. In other words

$$\hat{y} = g(\boldsymbol{\varphi}) \tag{4.2}$$

should be a good prediction of the actual output y. As real measurements will always suffer from noise, the regression problem can also be stated in a stochastic framework as the function $g(\boldsymbol{\varphi})$ should minimise e.g. the variance

$$E[y - g(\boldsymbol{\varphi})]^2. \tag{4.3}$$

4.1.2 Linear regression

In the special case of *linear regression* the regression function $g(\boldsymbol{\varphi})$ can be expressed *linearly in the parameters.* Note that this does *not* imply any linearity with respect to the regressors in $\boldsymbol{\varphi}$ or input x in Fig. 4.1 as will become clear in one of the examples.

Without loss of generality we will consider a special case in which the regression function can be written as

$$g(\boldsymbol{\varphi}) = \theta_1\varphi_1 + \theta_2\varphi_2 + ... + \theta_d\varphi_d. \tag{4.4}$$

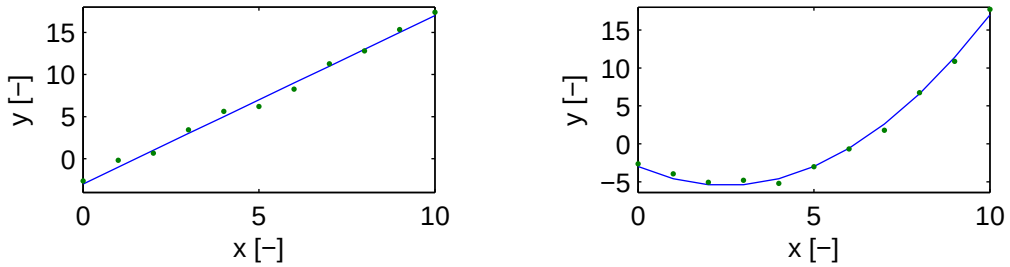
The parameters in this expression are

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_d \end{bmatrix}, \tag{4.5}$$

so in combination with Eq. (4.1) a shorthand notation for the regression function is

$$g(\boldsymbol{\varphi}) = \boldsymbol{\varphi}^T\boldsymbol{\theta}. \tag{4.6}$$

Examples



(a) (b)

Figure 4.2: Fitting data with (a) a linear and (b) a quadratic function.

To illustrate this notation, we consider the "well known" linear fit

$$y = ax + b, \tag{4.7}$$

see Fig. 4.2(a). Equation (4.6) can be applied with input vector

$$\boldsymbol{\varphi} = \begin{bmatrix} x \\ 1 \end{bmatrix}, \tag{4.8}$$

and parameter vector

$$\boldsymbol{\theta} = \begin{bmatrix} a \\ b \end{bmatrix}. \tag{4.9}$$

These vectors combine into the regression function

$$g(\boldsymbol{\varphi}) = \begin{bmatrix} x & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}, \tag{4.10}$$

which is clearly linear in the parameters a and b.

In this example the regression function depends also linearly on the input x. That is not the case for the quadratic function

$$y = c_2x^2 + c_1x + c_0, \tag{4.11}$$

see Fig. 4.2(b). Nevertheless, Eq. (4.6) can still be applied with with input vector

$$\boldsymbol{\varphi} = \begin{bmatrix} x^2 \\ x \\ 1 \end{bmatrix}, \tag{4.12}$$

and parameter vector

$$\boldsymbol{\theta} = \begin{bmatrix} c_2 \\ c_1 \\ c_0 \end{bmatrix} \tag{4.13}$$

These vectors combine into the regression function

$$g(\boldsymbol{\varphi}) = \begin{bmatrix} x^2 & x & 1 \end{bmatrix} \begin{bmatrix} c_2 \\ c_1 \\ c_0 \end{bmatrix}, \tag{4.14}$$

which is clearly linearly in the parameters. It is also linear in the regressors as in Eq. (4.4) although φ_1 is a non-linear function of the input x.

4.1.3 Least-squares estimate (LSE)

In practice we have to estimate the regression function $g(\boldsymbol{\theta})$ from a set with N measurements of related output and regressors

$$y(t), \boldsymbol{\varphi}(t), \qquad t = 1, ..., N, \tag{4.15}$$

respectively. The index t may refer to N independent measurements, but for dynamic systems it may also refer to actual time instances.

Next we need to define more precisely in what way the prediction $\hat{y}$ from the regression function (4.2) should "match" the actual output. Referring to Eq. (4.3) for the variance, a very often applied criterium is based on the *least squares error* that estimates this variance from the measured data as

$$V_N(\boldsymbol{\theta}) = \frac{1}{N} \sum_{t=1}^{N} [y(t) - g(\boldsymbol{\varphi}(t))]^2. \tag{4.16}$$

This so-called *cost function* is a function of the parameters $\boldsymbol{\theta}$. The "best" estimate $\hat{\boldsymbol{\theta}}_N$ for this parameter set is the one that minimises this cost function

$$\hat{\boldsymbol{\theta}}_N = \arg\min V_N(\boldsymbol{\theta}). \tag{4.17}$$

Normal equations

Substituting Eq. (4.6) for the linear case into Eq. (4.16) gives the cost function

$$V_N(\boldsymbol{\theta}) = \frac{1}{N} \sum_{t=1}^{N} [y(t) - \boldsymbol{\varphi}^T(t)\boldsymbol{\theta}]^2, \tag{4.18}$$

which is a quadratic function of $\boldsymbol{\theta}$. It can be minimised analytically as all partial derivatives

$$\frac{\partial V_N(\boldsymbol{\theta})}{\partial \theta_i}, \qquad (i = 1, 2, ..., d), \tag{4.19}$$

have to be zero in the minimum for which the parameter vector equals the best estimate $\hat{\boldsymbol{\theta}}_N$. That means that

$$\frac{1}{N} \sum_{t=1}^{N} 2\boldsymbol{\varphi}(t)[y(t) - \boldsymbol{\varphi}^T(t)\hat{\boldsymbol{\theta}}_N] = 0. \tag{4.20}$$

By rewriting this expression into the *normal equations*

$$\left[\frac{1}{N}\sum_{t=1}^{N}\boldsymbol{\varphi}(t)\boldsymbol{\varphi}^T(t)\right]\hat{\boldsymbol{\theta}}_N = \frac{1}{N}\sum_{t=1}^{N}\boldsymbol{\varphi}(t)y(t), \tag{4.21}$$

it becomes clear that for the linear case the "best fit" $\hat{\boldsymbol{\theta}}_N$ according to Eq. (4.17) is found as the solution of a set of N linear equation. Numerical procedures to compute this solution are readily available and furthermore the obtained minimum of the cost function is guaranteed to be a *global* minimum.

A requirement for the solution $\hat{\boldsymbol{\theta}}_N$ to exist, is that the matrix on the left in Eq. (4.21) is invertible. Then the *Least Squares Estimate* (LSE) can be written as

$$\hat{\boldsymbol{\theta}}_N = \left[\frac{1}{N}\sum_{t=1}^{N}\boldsymbol{\varphi}(t)\boldsymbol{\varphi}^T(t)\right]^{-1}\frac{1}{N}\sum_{t=1}^{N}\boldsymbol{\varphi}(t)y(t). \tag{4.22}$$

Matrix formulation

A shorthand notation for Eq. (4.22) can be obtained by collecting the output measurements in the output vector

$$Y_N = \begin{bmatrix} y(1) \\ \vdots \\ y(N) \end{bmatrix}, \tag{4.23}$$

and the input regressors in the $N \times d$ regression matrix

$$\Phi_N = \begin{bmatrix} \boldsymbol{\varphi}^T(1) \\ \vdots \\ \boldsymbol{\varphi}^T(N) \end{bmatrix}. \tag{4.24}$$

The cost function of Eq. (4.18) can then be written as

$$V_N(\boldsymbol{\theta}) = \frac{1}{N}||Y_N - \Phi_N\boldsymbol{\theta}||_2^2, \tag{4.25}$$

in which $||.||_2$ denotes the 2-norm of a vector. In this notation the normal equations (4.21) are (after multiplication with N) expressed as

$$\left[\Phi_N^T\Phi_N\right]\hat{\boldsymbol{\theta}}_N = \Phi_N^T Y_N, \tag{4.26}$$

and the LSE of Eq. (4.22) becomes

$$\hat{\boldsymbol{\theta}}_N = \Phi_N^\dagger Y_N \tag{4.27}$$

in which $\Phi_N^\dagger$ denotes the (Moore-Penrose) *pseudoinverse* of Φ_N

$$\Phi_N^\dagger = \left[\Phi_N^T\Phi_N\right]^{-1}\Phi_N^T. \tag{4.28}$$

For a square invertible matrix the pseudoinverse equals the inverse. In general it is true that

$$\Phi_N^\dagger\Phi_N = I, \tag{4.29}$$

where the dimensions of the identify matrix I match the product on the left ($d \times d$).

LSE in MATLAB

The LSE is an example of the solution of an overdetermined set of equations, i.e. the regression matrix Φ_N has more rows than columns ($N > d$). For this estimate a number of numerical solutions are available in MATLAB. Suppose that the regression matrix Φ_N is stored in variable `PHIN` and the output vector Y_N is available as `YN`. The preferred solution in many cases is

```
theta = PHIN\YN;                    % Preferred
```

for which a numerically sound algorithm is used that is efficient when only one solution has to be computed. An alternative is the use of the `pinv` command that actually computes the pseudoinverse

```
theta = pinv(PHIN)*YN;              % Alternative
```

Note that a "user implementation" of Eq. (4.28)

```
theta = inv(PHIN'*PHIN)*PHIN'*YN; % Not favourable
```

should not be used as the computation of the pseudoinverse $\Phi_N^\dagger$ and the solution $\hat{\boldsymbol{\theta}}_N$ this way are less accurate, see e.g. the MATLAB documentation for the `inv`, `pinv` and `mldivide` commands.

Examples

Suppose we would like to apply the linear fit

$$y = ax + b, \tag{4.30}$$

for a set with N measurements x_i and y_i ($i = 1, ..., N$). Therefore, we would like to minimise the cost function

$$V_N = \frac{1}{N}\sum(y_i - ax_i - b)^2, \tag{4.31}$$

which is a function of the parameter vector in Eq. (4.9).

First we will compute the LSE "manually" from the partial derivatives of the cost function that have to satisfy

$$\frac{\partial V_N}{\partial a} = 0 \quad \text{and} \quad \frac{\partial V_N}{\partial b} = 0. \tag{4.32}$$

This leads to

$$\left\{ \begin{array}{rcl} \sum -x_i(y_i - ax_i - b) & = & 0 \\ \sum -(y_i - ax_i - b) & = & 0 \end{array} \right. \Leftrightarrow \left[\begin{array}{cc} \sum x_i^2 & \sum x_i \\ \sum x_i & \sum 1 \end{array} \right] \left[\begin{array}{c} a \\ b \end{array} \right] = \left[\begin{array}{c} \sum x_i y_i \\ \sum y_i \end{array} \right] \tag{4.33}$$

from which the LSE is found as

$$\left[\begin{array}{c} \hat{a} \\ \hat{b} \end{array} \right] = \left[\begin{array}{cc} \sum x_i^2 & \sum x_i \\ \sum x_i & \sum 1 \end{array} \right]^{-1} \left[\begin{array}{c} \sum x_i y_i \\ \sum y_i \end{array} \right] \tag{4.34}$$

Although in this case the computation of the solution is quite straightforward, a drawback of this approach is that the derivation has to be repeated for every change in the regression function. Instead we can also try to express the problem in the matrix formulation for which the solution is immediately found with the pseudoinverse.

For the matrix formulation, we need to collect the outputs into

$$Y_N = \left[\begin{array}{c} y_1 \\ \vdots \\ y_N \end{array} \right]. \tag{4.35}$$

To compute the regression matrix Φ_N we substitute the expression for the input vector Eq. (4.8) into Eq. (4.24) yielding

$$\Phi_N = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \tag{4.36}$$

Then, the LSE solution for the parameter vector Eq. (4.9)

$$\hat{\boldsymbol{\theta}}_N = \begin{bmatrix} \hat{a} \\ \hat{b} \end{bmatrix} = \Phi_N^{\dagger} Y_N. \tag{4.37}$$

As a second example we consider the pre-whitening filter that was used for the estimate of the impulse response and introduced in Eq. (3.14) according to

$$L(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + ... + a_n z^{-n}. \tag{4.38}$$

This linear filter of fixed order n (e.g. $n = 10$) should be applied to a filter signal $u(k)$ into

$$u_F(k) = L(z)u(k), \tag{4.39}$$

such that the cost function

$$V_N = \frac{1}{N} \sum_{k=1}^{N} u_F^2(k) \tag{4.40}$$

is minimised. For a "manual" solution we would (again) have to set all partial derivatives

$$\frac{\partial V_N}{\partial a_i} = 0 \tag{4.41}$$

for all $i = 1, 2, ..., n$ and solve the linear set of equations. Instead we will apply the matrix formulation by rewriting the cost function in vector form of Eq. (4.25) for which the LSE can be computed directly with Eq. (4.27). Therefore we rewrite $u_F(t)$ as

$$u_F(k) = u(k) + a_1 u(k-1) + a_2 u(k-2) + ... + a_n u(k-n), \tag{4.42}$$

and substitute this expression into Eq. (4.40) to obtain

$$V_N = \frac{1}{N} \left\| \begin{pmatrix} u(n+1) + a_1 u(n) + a_2 u(n-1) + ... + a_n u(1) \\ u(n+2) + a_1 u(n+1) + a_2 u(n) + ... + a_n u(2) \\ \vdots \\ n(N) + a_1 u(N-1) + a_2 u(N-2) + ... + a_n u(N-n) \end{pmatrix} \right\|_2^2 . \tag{4.43}$$

By comparing this expression with Eq. (4.25) it is possible to recognise Y_N and Φ_N and next to compute the best fit for the parameter vector

$$\boldsymbol{\theta} = [a_1 a_2 ... a_n]^T. \tag{4.44}$$

For a third example we consider the fit of a dynamic system with a so-called ARX model. Such models will be discussed in detail in Sect. 6.3. For the moment we will introduce the equations without further explanation. In an ARX model the relation between input u_k and output y_k is written as

$$y_k = \frac{B(z)}{A(z)} u_k + \frac{1}{A(z)} e_k, \tag{4.45}$$

in which e_k is a white noise disturbance and $A(z)$ and $B(z)$ are polynomials of fixed order of which the coefficients should be determined from N measurements of u_k and y_k ($k = 1, ..., N$). Consider e.g.

$$A(z) = 1 + a_1 z^{-1}, \tag{4.46}$$

and

$$B(z) = b_1 z^{-1} + b_2 z^{-2}. \tag{4.47}$$

Then

$$(1 + a_1 z^{-1}) y_k = (b_1 z^{-1} + b_2 z^{-2}) u_k + e_k, \tag{4.48}$$

so the output at time instant k can be computed with

$$y_k = -a_1 y_{k-1} + b_1 u_{k-1} + b_2 u_{k-2} + e_k, \tag{4.49}$$

from past inputs, outputs and the (unknown) disturbance e_k. The ARX model can not take this (unknown) disturbance e_k into account and the model has to be estimated from the available input and output data. Then a prediction for the output follows from Eq. (4.49) by discarding the disturbance

$$\hat{y}_k = -a_1 y_{k-1} + b_1 u_{k-1} + b_2 u_{k-2}. \tag{4.50}$$

The cost function to evaluate this estimate is

$$V_N = \frac{1}{N} \sum_{k=1}^{N} (y_k + a_1 y_{k-1} - b_1 u_{k-1} - b_2 u_{k-2})^2, \tag{4.51}$$

which depends on the parameter vector

$$\boldsymbol{\theta} = \begin{bmatrix} a_1 \\ b_1 \\ b_2 \end{bmatrix}. \tag{4.52}$$

The cost function can be written in the matrix formulation by collecting the output vector

$$Y_N = \begin{bmatrix} y(3) \\ \vdots \\ y(N) \end{bmatrix} \tag{4.53}$$

and recognising the regression matrix

$$\Phi_N = \begin{bmatrix} -y(2) & u(2) & u(1) \\ \vdots & \vdots & \vdots \\ -y(N-1) & u(N-1) & u(N-2) \end{bmatrix}. \tag{4.54}$$

Then obviously the LSE follows once more from Eq. (4.27).

4.1.4 Summary

To summarise this section, we note that in the case the model of our system is linear in the parameters, the Least Squares Estimate (LSE) for the parameters can be computed easily from the normal equations:

- Suppose we can write the cost function as

$$V_N = \frac{1}{N} ||Y_N - \Phi_N \boldsymbol{\theta}||_2^2.$$

- Then the LSE can be computed from the normal equations

$$\left[\Phi_N^T \Phi_N\right] \hat{\boldsymbol{\theta}}_N = \Phi_N^T Y_N.$$

- The solution for this best fit is

$$\hat{\boldsymbol{\theta}}_N = \Phi_N^\dagger Y_N,$$

with the pseudo-inverse $\Phi_N^\dagger = \left[\Phi_N^T \Phi_N\right]^{-1} \Phi_N^T$.

For a lot of cases that is all that is needed to obtain a satisfying solution. However, there are a number of pitfalls and complications that may have to be taken care of:

- Analytical difficulty: What if $\left[\Phi_N^T \Phi_N\right]$ is singular? The pseudoinverse does not exist.
- Numerical difficulty: What if $\left[\Phi_N^T \Phi_N\right]$ is near-singular? The pseudoinverse is defined mathematically, but it is not possible to compute it accurately. What does that mean and how are the normal equations solved anyhow (e.g. by MATLAB)?
- How about the accuracy of the estimate $\hat{\boldsymbol{\theta}}_N$?

The answers to these questions can be given for specific problems like the linear fit for $y = ax + b$. Instead, we will investigate these matters by considering the *Singular Value Decomposition* (SVD) in the next section.

4.2 LSE and Singular Value Decomposition

The *Singular Value Decomposition* (SVD) of any $N \times d$ matrix Φ with N (rows) $> d$ (columns), like the regression matrix, can be written as

$$\Phi = U\Sigma V^T \quad \text{or} \quad \begin{bmatrix} \\ \Phi \\ \\ \end{bmatrix} = \begin{bmatrix} \\ U \\ \\ \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_d \\ & 0 & \end{bmatrix} \begin{bmatrix} \\ V^T \\ \\ \end{bmatrix}, \tag{4.55}$$

in which U is an orthogonal $N \times N$ matrix and V is an orthogonal $d \times d$ matrix. Matrix Σ is a $N \times d$ matrix with the *singular values*

$$\Sigma_{ii} = \sigma_i \geq 0, \qquad \text{(i=1,2,...,d)} \tag{4.56}$$

on the diagonal of the upper part and 0 elsewhere. Usually the singular values are sorted:

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_d \geq 0. \tag{4.57}$$

Note that only the left d columns of matrix U contribute to the matrix product as all elements in the lower part of Σ are zero.

The Singular Value Decomposition provides a well-defined way to detect singularity or near-singularity by means of the condition number of the matrix Φ. This condition number is defined as the ratio of the largest and smallest singular values

$$c(\Phi) = \frac{\sigma_1}{\sigma_d}. \tag{4.58}$$

The following cases for the condition number can be distinguished:

- If all $\sigma_i > 0$ ($i = 1, 2, ..., d$), then the condition number is finite. The matrix $\left[\Phi^T\Phi\right]$ is invertible. Matrix Φ has full (column) rank.
- If $\sigma_i = 0$ for $i = r+1, r+2, ..., d$, the condition number is infinite and matrix $\left[\Phi^T\Phi\right]$ is singular. The number of non-zero singular values r is the rank of matrix Φ.
- "Large" condition numbers indicate near-singularity of matrix $\left[\Phi_N^T\Phi_N\right]$ and will lead to numerical inaccuracies.

This will be illustrated with a number of examples.

Consider quadratic function introduced before in Eq. (4.11)

$$y = c_2x^2 + c_1x + c_0. \tag{4.59}$$

It is easy to show that the regression matrix can be written as

$$\Phi_N = \begin{bmatrix} x^2(1) & x(1) & 1 \\ \vdots & \vdots & \vdots \\ x^2(N) & x(N) & 1 \end{bmatrix}, \tag{4.60}$$

which obviously only depends on the input data $x(i)$.

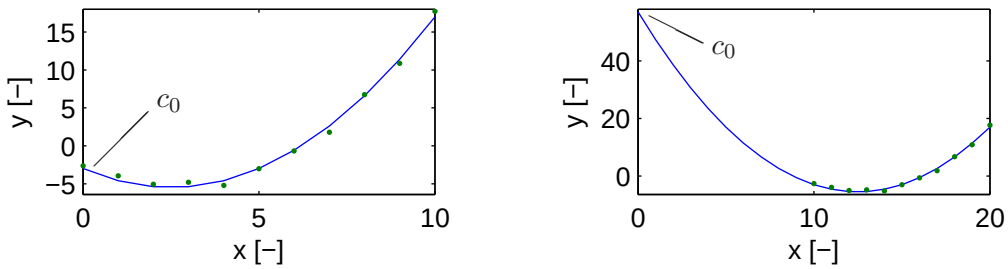


Figure 4.3: Fitting data with a quadratic function. Plot (a) represents the original data as in Fig. 4.2(b). In (b) the data are shifted in horizontal direction.

Figure 4.3 shows two examples of data sets that are generated for this function. These data sets are almost identical except that in the right graph all values of the input data $x(i)$ are increased with a constant value of 10.

Computing the singular values of the regression matrix, the condition number and the best parameter set gives for the left and right graph respectively:

`svd(phi)` = $[160.3;\ 5.2;\ 1.2]$	`svd(phi)` = $[842.5;\ 9.1;\ 0.1]$
`cond(phi)` = 131	`cond(phi)` = 6364
`phi\y` = $[0.43;\ -2.33;\ -2.25]$	`phi\y` = $[0.43;\ -10.9;\ 63.8]$
Exact: $[0.40;\ -2.00;\ -3.00]$	Exact: $[0.40;\ -10.0;\ 57.0]$

The condition number indicated that quality of the fit for the data of the right graph is less. From the estimated parameters it appears that in particular the third parameter, which is the offset c_0 along the vertical axis, has a rather large (absolute) error in the right graph. As illustrated in the figure, that is not so surprising, as this parameter is determined by a rather large extrapolation with respect to the available data.

For a second example we reconsider the data in Fig. 4.2(a) (page 4-2) that should satisfy the "well known" linear fit of Eq.(4.7)

$$y = ax + b. \tag{4.61}$$

Alternatively, it will be attempted to fit this data with

$$y = a_1 x + a_2(10 - x) + b. \tag{4.62}$$

The regression matrices for these fits are respectively

$$\Phi_N = \begin{bmatrix} x(1) & 1 \\ \vdots & \vdots \\ x(N) & 1 \end{bmatrix}, \tag{4.63}$$

and

$$\Phi_N = \begin{bmatrix} x(1) & 10\text{–}x(1) & 1 \\ \vdots & \vdots & \vdots \\ x(N) & 10\text{–}x(N) & 1 \end{bmatrix}. \tag{4.64}$$

The analysis for either fit gives

`svd(phi)` = $[19.8;\ 1.75]$	`svd(phi)` = $[23.7;\ 14.8;\ 0.0]$
`cond(phi)` = 11.3	`cond(phi)` = $5.7 \cdot 10^{16}$
`phi\y` = $[1.97;\ -2.76]$	`phi\y` = $[1.70;\ -0.28; 0]$
	Warning: Rank deficient, rank = 2 tol = 4.8e-14.
Exact: $[2.00;\ -3.00]$	Exact: $[2.00;\ 0.00;\ -3.00]$

In the right fit the smallest singular value is (almost) zero and hence the condition number is large. Consequently, not all parameters can be estimated from the data and indeed only two parameters are actually estimated. This difficulty can be understood easily by rewriting Eq. (4.62) as

$$y = (a_1 - a_2)x + (10a_2 + b), \tag{4.65}$$

from which it is clear that any set of parameters with equal $a_1 - a_2$ and $10a_2 + b$ yield the same output y for a given input x. The three parameters can not be determined uniquely which is obvious from the fit equations in this simple example. It can also be concluded quite easily from the regression matrix (4.64) as the middle column can be written as a linear combination of the other two columns. Yet another indication that the regression matrix is not full rank.

For a simple example like this, the singular value decomposition is not needed to understand and resolve the ambiguity of the fit problem. We will see later that for more complicated model equations such column dependencies may be much more difficult to observe and then singular value decomposition may be a very valuable tool to detect rank deficiency. Furthermore, it also provides a clear solution for a (near) singular regression matrix.

To illustrate that, we will first split the decomposition of Eq. (4.55) into two parts

$$\begin{bmatrix} \\ \Phi \\ \\ \end{bmatrix} = \begin{bmatrix} & & \\ U_1 & U_2 & \\ & & \end{bmatrix} \begin{bmatrix} \Sigma_1 & \\ & \Sigma_2 \\ 0 & \end{bmatrix} \begin{bmatrix} & \\ V_1 & V_2 \\ & \end{bmatrix}^T, \tag{4.66}$$

in which the r non-zero singular values are collected in Σ_1 and $\Sigma_2 \approx 0$. In other words, Σ_1 is a $r \times r$ diagonal matrix with the nonzero singular values $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0$. The rest of the matrix, including Σ_2, is zero. When Φ is a $N \times d$ matrix with more rows than columns ($N > d$), then U is an orthogonal $N \times N$ matrix of which U_1 are the first r columns and V is an orthogonal $d \times d$ matrix of which V_1 are the first r columns. Only U_1, Σ_1 and V_1 contribute to Φ, so we can also write

$$\Phi = U_1 \Sigma_1 V_1^T \tag{4.67}$$

or

$$\begin{bmatrix} \\ \Phi \\ \\ \end{bmatrix} = \begin{bmatrix} \\ U_1 \\ \\ \end{bmatrix} \begin{bmatrix} \\ \Sigma_1 \\ \\ \end{bmatrix} \begin{bmatrix} \quad V_1^T \quad \end{bmatrix}. \tag{4.68}$$

Next we compute the (pseudo) inverse by only considering this part, so

$$\Phi^\dagger = V_1 \Sigma_1^{-1} U_1^T, \tag{4.69}$$

where the inverse of Σ_1 is a diagonal matrix with the inverse non-zero singular values on the diagonal

$$\Sigma_1^{-1} = \text{diag}(1/\sigma_1, 1/\sigma_2, ..., 1/\sigma_r). \tag{4.70}$$

Finally, we can use this matrix to compute one solution of the least squares fit, namely

$$\hat{\boldsymbol{\theta}}_N = V_1 \Sigma^{-1} U_1^T \, Y_N. \tag{4.71}$$

This is actually the output of the MATLAB command `pinv(phi)*yn`

As pointed out above, this solution is not unique when the rank r of Φ is less than d. The matrix V_2 can be used to find all solutions. That is easy to show as Σ_2 is zero, so any linear combination of the columns of V_2 can be added to the parameter vector without contributing to the regression matrix.

Returning to the data in Fig. 4.2(a) and the fit of Eq. (4.62) with the regression matrix according to Eq. (4.64), it was found that one singular value was indeed (almost) equal to zero. So the rank of the regression matrix is 2 and matrix V_2 has one column. Instead of using the `pinv` command directly, we will compute the solution "manually":

```
[u,s,v]=svd(phi);
r=1:2;                              % rank
u1=u(:,r); s1=s(r,r); v1=v(:,r);
% Compute pinv(phi)*y:
v1*diag(1./diag(s1))*u1' * y =
            [ 1.68; -0.29; 0.14 ]
v2=v(:,3) = [ 0.099; 0.099; -0.99 ]
```

The solution obtained this way differs from the previously computed best fit. In addition the matrix V_2 is computed and now we can determine all possible best solutions. E.g. once the reason for the singularity is clear, it is obvious that the original linear fit Eq. (4.61) should be used to fit the data. In other words, the parameter a_2 can taken zero. This can be accomplished by adding the single column of matrix V_2 about three times to the solution of the fit. Then the solution becomes $[1.97;\ 0;\ -2.76]$, which is identical the result that was found when the correct fit equation had been applied from the start.

In this first discussion about the estimation of parameters from data we have seen that the Singular Value Decomposition (svd) can provide the numerical tools to compute the linear least squares estimate. Furthermore it can detect a (near) singular regression matrix. The pseudo inverse of the regression matrix Φ_N can always be computed reliably using the Singular Value Decomposition. When the rank of the regression matrix equals r, then r parameters or linear combinations of parameters are obtained. In case of non-uniqueness all solutions can be found found by taking into account matrix V_2.

So far, the accuracy of the estimate $\hat{\boldsymbol{\theta}}_N$ has not been discussed in full detail. This will be considered in one of the future chapters.

5 Realisation algorithms

5.1 Introduction

In this chapter we will discuss the so-called realisation algorithms, namely the *approximate realisations* and the *subspace identifications* techniques. Both are example of parametric identification techniques, so they estimate parametric models with a limited number of mathematical parameters.

The approximate realisations are discussed first. They can be introduced easily by combining knowledge of the impulse response of a system with a computation of the rank and a decomposition of a matrix, e.g. by means of singular value decomposition. These techniques are not available in the `ident` toolbox, so for practical applications the subspace techniques are more favourable. These identification techniques are applied directly to input-output data without the explicit need of an impulse response. The mathematical background of these techniques is more complicated and will only be outlined briefly.

5.2 Approximate realisations

Suppose we know the impulse response $g(k)$ for $k = 0, 1, ..., \infty$ (the *Markov parameters*) of a finite-dimensional linear time invariant (LTI) system. We would like to have a state space model of minimal order for this system, like

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k) \end{aligned} \tag{5.1}$$

We note that both the impulse response and the state space matrices are related to the system's transfer function $G(z)$. One way to express the transfer function is

$$G(z) = \sum_{k=0}^{\infty} g(k) z^{-k}. \tag{5.2}$$

Secondly, it can be expressed in the state matrices of Eq. (5.1) as

$$G(z) = D + C(zI - A)^{-1}B. \tag{5.3}$$

Unfortunately, these expressions do not show immediately how the state space matrices can be computed from the impulse response. The other way is rather easy as it can be seen from Eq. (5.1) that

$$g(k) = \begin{cases} D & k = 0, \\ CA^{k-1}B & k \geq 1. \end{cases} \tag{5.4}$$

In the case the system shows *direct feedthrough*, matrix D is non-zero and its value follows directly from the impulse response $g(0)$, so the output at the instant $t = 0$ when the impulse is applied.

In 1966 Ho and Kalman [11] showed a way to find the matrices A, B and C from the rest of the impulse response. First they compose the so-called Hankel matrix H of dimension $n_r \times n_c$, which is

defined as

$$H_{n_r,n_c} = \begin{bmatrix} g(1) & g(2) & g(3) & \cdots & g(n_c) \\ g(2) & g(3) & g(4) & \cdots & g(n_c+1) \\ g(3) & g(4) & g(5) & \cdots & g(n_c+2) \\ \vdots & \vdots & \vdots & & \vdots \\ g(n_r) & g(n_r+1) & g(n_r+2) & \cdots & g(n_r+n_c-1) \end{bmatrix}. \tag{5.5}$$

Using Eq. (5.4) it follows that the Hankel matrix can also be expressed in the state matrices according to

$$H_{n_r,n_c} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n_r-1} \end{bmatrix} \begin{bmatrix} B & AB & \cdots & A^{n_c-1}B \end{bmatrix}. \tag{5.6}$$

Readers familiar with control theory may recognise this as the product between the observability matrix and controllability matrix. What follows it that the rank of the Hankel matrix equals the order n of the system matrix A, provided that n_r and n_c are taken sufficiently large. In other words, the (minimal) order n of the system follows directly from the rank of the Hankel matrix. What then remains is to determine the rank and find a way to compute the matrices.

A reliable numerical computation of the rank of the matrix was far from trivial at the moment Ho and Kalman published their algorithm. In 1974 singular value decomposition (SVD) was proposed as an elegant solution for this problem [12]. We use SVD to write the Hankel matrix as

$$H = U_n \Sigma_n V_n^T, \tag{5.7}$$

with U_n and V_n unitary matrices ($U_n^T U_n = V_n^T V_n = I_n$) and Σ_n a diagonal matrix with the positive singular values

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n > 0 \tag{5.8}$$

on the diagonal. As zero singular values are not taken into account in this singular value decomposition, the number of singular values n equals the rank of the Hankel matrix and hence also the (minimal) order of the system matrix A.

Furthermore, we write

$$H = U_n \Sigma_n^{1/2} \quad \Sigma_n^{1/2} V_n^T, \tag{5.9}$$

in which the notation $\Sigma_n^{1/2}$ indicates a diagonal matrix with the square roots of the singular values on the diagonal. Then the input and output matrices in the state space equations can be written as

$$B\text{: the first column of } H_2 = \Sigma_n^{1/2} V_n^T, \tag{5.10}$$

and

$$C\text{: the first row of } H_1 = U_n \Sigma_n^{1/2}. \tag{5.11}$$

The system matrix is then

$$A = \Sigma_n^{-1/2} U_n^T \quad \overleftarrow{H} \quad V_n \Sigma_n^{-1/2}, \tag{5.12}$$

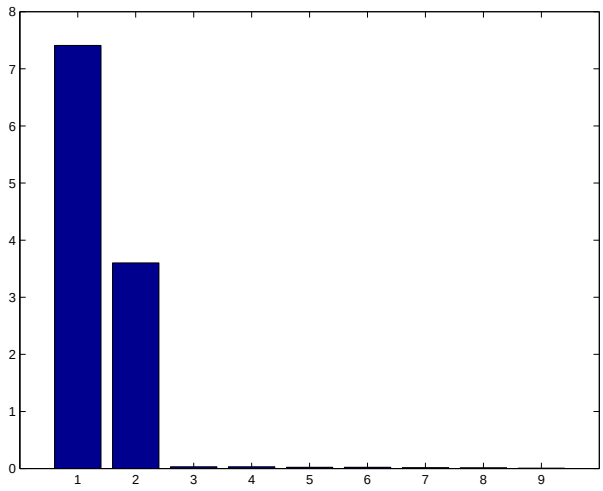
where the *shifted* Hankel matrix is defined as

$$\overleftarrow{H}_{n_r,n_c} = \begin{bmatrix} g(2) & g(3) & g(4) & \cdots & g(n_c+1) \\ g(3) & g(4) & g(5) & \cdots & g(n_c+2) \\ g(4) & g(5) & g(6) & \cdots & g(n_c+3) \\ \vdots & \vdots & \vdots & & \vdots \\ g(n_r+1) & g(n_r+2) & g(n_r+3) & \cdots & g(n_r+n_c) \end{bmatrix}. \tag{5.13}$$

For real data there will always be noise and $g(k)$ is only a finite series for $k = 0, 1, ..., N$. The proposed procedure in that case is

- Construct a (sufficiently large) Hankel matrix H with $n_c + n_r = N$.
- Apply SVD ($H = U\Sigma V^T$) and find out how many n singular values are *significantly* nonzero.
- Next construct an approximating Hankel matrix of rank n according $H_n = U_n\Sigma_n V_n^T$, in which U_n and V_n are the first n columns of U and V. Σ_n is the diagonal matrix with the first n (nonzero) singular values.
- Apply the original Ho&Kalman algorithm to H_n to compute B and C. For A the shifted Hankel matrix $\overleftarrow{H}$ with the original series $g(k)$ is used.

Example:



SVD's:
7.410
3.602
0.033
0.033
0.023
0.023
0.016
0.015
0.009

Figure 5.1: Plot and values of the singular values of the Hankel matrix constructed from the impulse response of a second order system.

To illustrate this procedure, we consider the estimated impulse (with `cra`) of simulated data from a second order system with transfer function

$$G_0(z) = \frac{z^{-1} + 0.5z^{-2}}{1 - 1.5z^{-1} + 0.7z^{-2}}. \tag{5.14}$$

Figure 5.1 shows the singular values of the Hankel matrix. Apparently only two singular values are distinct from zero and hence the order of the system appears indeed to be $n = 2$. Applying the rest of the procedure gives for the matrices

$$\begin{aligned} A &= \begin{bmatrix} 0.829 & -0.379 \\ 0.379 & 0.673 \end{bmatrix}, & B &= \begin{bmatrix} 1.392 \\ -0.968 \end{bmatrix}, \\ C &= \begin{bmatrix} 1.3915 & 0.9681 \end{bmatrix}, & D &= \begin{bmatrix} 0 \end{bmatrix}. \end{aligned} \tag{5.15}$$

Of course, these matrices are not unique as the states x_k are not uniquely defined. Nevertheless, the transfer function can be computed easily from these matrices and Eq. (5.3) and appears to be

$$\hat{G}(z) = \frac{0.999z^{-1} + 0.496z^{-2}}{1 - 1.502z^{-1} + 0.701z^{-2}} \tag{5.16}$$

which is close to the original $G_0(z)$. The agreement is confirmed by the pole-zero plot of both the exact system and the estimate, Fig. 5.2.

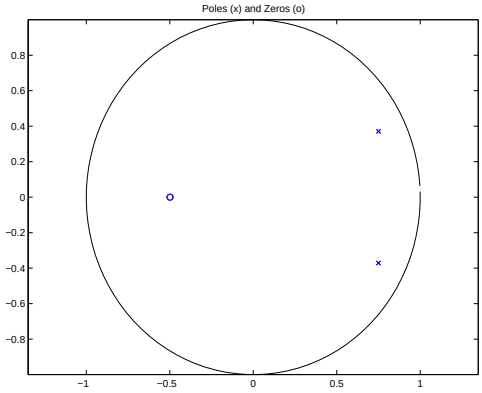


Figure 5.2: Pole zero maps of a true second order system G_0 and the estimate $\hat{G}$.

5.3 Subspace identification

A drawback of the approximate realisation discussed in the previous section is that its input is the impulse response. In practice, this impulse response in quite likely identified from input-output data. This intermediate result is not needed in more recent developed realisation methods, the so-called subspace identification of which some of the algorithms originate from research carried out in the early 1990's at the Catholic University of Leuven (KUL), see e.g. [13, 14]. Nowadays, subspace identification algorithms are e.g. provided by the `n4sid` command as part of the `ident` toolbox.

It has some advantages and disadvantages compared to e.g. the so-called *prediction error methods* that will be discussed in the next chapter. An advantage is that the applied model structure of a state space system has no need for a more explicit definition of the model equations. One has only to select the order of the system matrix. In addition the model structure is well suited for MIMO systems. Furthermore the numerical and mathematical approach is elegant (robust, reliable) and efficient (optimisation with linear equations). Note however, that not all mathematical issues of the optimisation are settled yet. As the model structure offers little means to apply dedicated model equations, it may appear that other techniques yield "better" system estimates, i.e. the subspace model is "sub-optimal'. In many cases the subspace model may be good enough for the intended application. When more advanced, but also numerically more complicated, algorithms are preferred, the sub-space model may still be very valuable to provide an initial guess.

A typical state space model structure for a system including a noise source v_k can be written as

$$\begin{array}{rcll} x_{k+1} & = & A\,x_k + B\,u_k & \text{Measured input and output } u_k \text{ and } y_k, \\ y_k & = & C\,x_k + D\,u_k + v_k & \text{Noise source } v_k. \end{array} \tag{5.17}$$

For the identification of the system matrices from measured data, we need do characterise the noise source in more detail. In the *innovations* form the equations are written as

$$\begin{array}{rcll} x_{k+1} & = & A\,x_k + B\,u_k + K\,e_k & \text{Measured input and output } u_k \text{ and } y_k, \\ y_k & = & C\,x_k + D\,u_k + e_k & \text{White noise source } e_k. \end{array} \tag{5.18}$$

In these equations the noise source e_k is white and the matrix K plays a role to determine the spectrum of the noise. In Chapter 6 we will frequently use the expression

$$y_k = G(z)u_k + H(z)e_k, \tag{5.19}$$

to explicitly specify a system model $G(z)$ and noise model $H(z)$. They are related to the system matrices according to

$$G(z) = C(zI - A)^{-1}B + D, \tag{5.20}$$

and

$$H(z) = C(zI - A)^{-1}K + I. \tag{5.21}$$

For the subspace identification we would like to estimate the system matrices in Eq. (5.18) from measured input and output data, u_k and y_k respectively. If also the states were also known, then the solution would be straightforward: Compute C and D with linear regression, reconstruct e_k, compute A, B and K also with linear regression. Now the "trick" of the subspace identification is to find the states x_k. An important aspect of this problem is to determine the number of states, so the order n of the system.

An overview of subspace techniques is given by Ljung [6, Section 10.6]. An essential step is the reconstruction of the (extended) observability matrix (see also Eq. (5.6))

$$O_r = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{r-1} \end{bmatrix} \tag{5.22}$$

from input-output data. Again the rank of this matrix equals the order n of the system and Singular Value Decomposition can be applied once more for this purpose. The plot shown in `ident`'s "Order selection" reflects the singular values of a (weighted) estimate of this matrix. Once the rank and the observability matrix are known, it is easy to determine C and A. Output matrix C equals the top part of this matrix and this result can be used to compute A from the rest. Furthermore, the states x_k and the noise contributions can be estimated. The matrices B and D as well as the initial state x_0 are estimated from a linear regression problem. For more details the interested reader is referred to Ljung [6].

Examples:

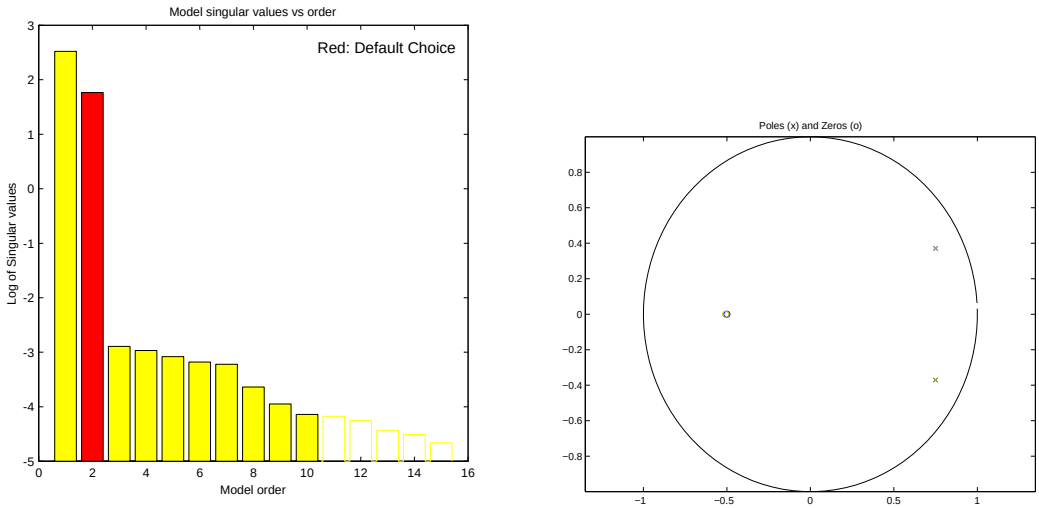


Figure 5.3: Subspace identification applied to simulated data from a second order system. The left graph shows the singular values to estimate the order n. The right graph shows the pole-zero map for the real system G_0 and the second order estimate $\hat{G}_{\mathrm{n4s2}}$.

The application of the subspace identification will be illustrated with simulated and real data. First simulated data is once more generated with the second order system of Eq. (5.14). The left graph in Fig. 5.3 shows the singular values according to `ident`'s "Order selection" from which it is obvious that the system is second order. The pole-zero map of the estimated second order subspace model, denoted `n4s2`, is compared with the known model in the right graph. The agreement is clear.

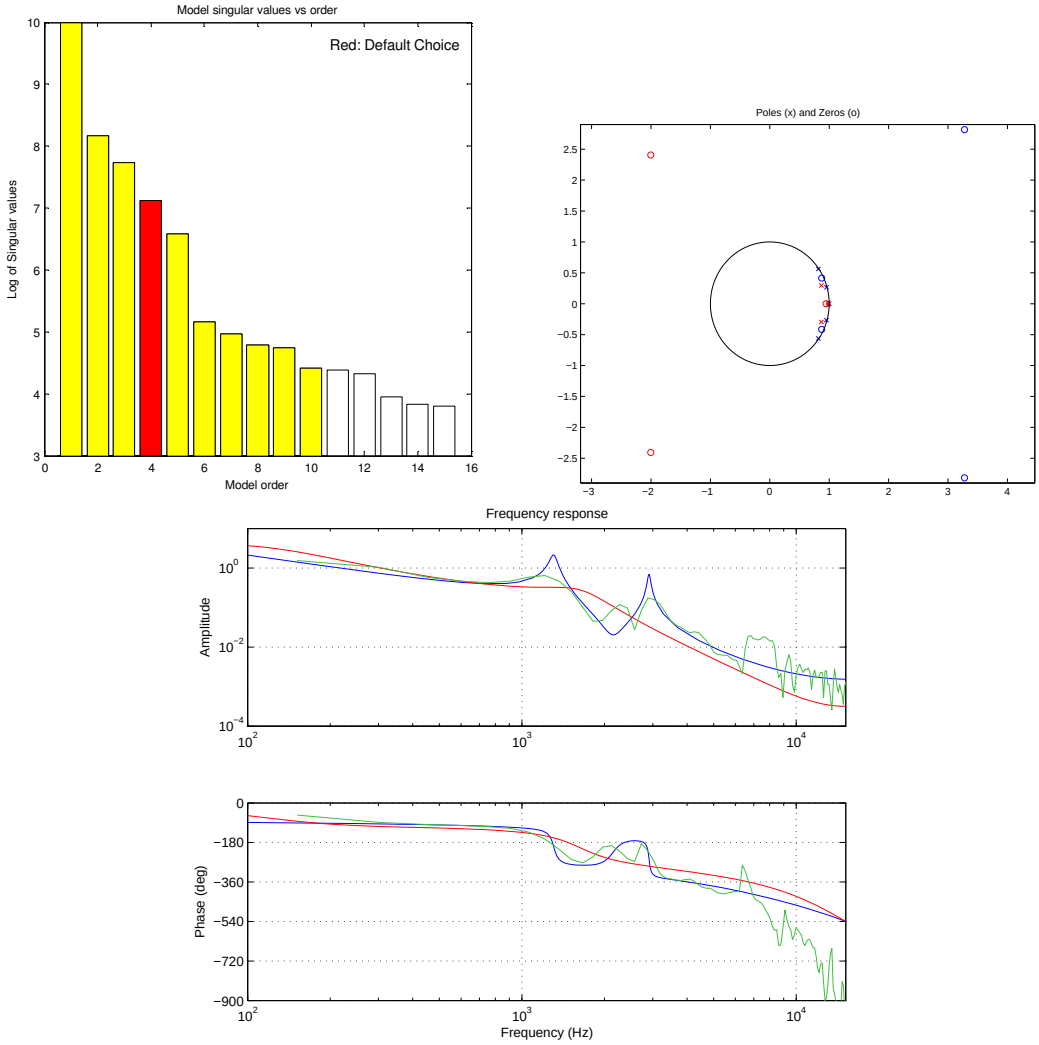


Figure 5.4: Subspace identification applied to the measured data of a piezo mechanism. The left graph shows the singular values to estimate the order n. The right graph shows the pole-zero map for the estimates $\hat{G}_{\mathrm{n4s4}}$ and $\hat{G}_{\mathrm{n4s5}}$. The lower graph shows the spectral analysis of these models combined with a non-parametric ETFE60-model.

Next the procedure is applied to measured data from the piezo mechanism, Appendix A.3. In the graph with the singular values of Fig. 5.4 `ident`'s default choice is indicated by the red bar: A fourth order system. However, this example shows that this choice is not always the best. Already from the singular values one may argue that the difference between the fifth and sixth singular values is larger than between the fourth and fifth. This would suggest a *fifth* order model.

The pole-zero map of the models shows another shortcoming of the fourth order model: Near $+1$ there are one zero and two poles quite close to each other, indication pole-zero cancellation. So the fourth pole is not used efficiently to model system dynamics. The difference between the fourth order estimate `n4s4` and the fifth order model `n4s5` is perhaps best illustrated with the spectral analysis in Fig. 5.4. Of course there is no known system in this case, but a non-parametric ETFE can provide a reliable estimate. Comparing the estimated subspace models with the ETFE spectrum, it becomes clear that the fifth order model presents a reasonable guess of the frequency response up to about the third eigenfrequency. The first resonance is estimated correctly, but as the order of the subspace model is too low to capture all three resonances, the second and third resonances are combined somehow. It is up to the user to decide whether this model is accurate enough or that a higher order is required. Note finally, that the damping of the first resonance is significantly lower in the estimated subspace model than in the

ETFE model in which some smoothing is applied to reduce the fluctuations in the graph.

This discussion of the estimated models clearly illustrates that we need means to *validate* the estimated models and to be able to select the "best model" in the end. We will discuss this topic in much more detail in the next chapter. For the moment we note that a non-parametric spectral model may serve very well as a reference for the parametric models.

6 Prediction Error identification Methods

6.1 Introduction

There may be several reasons for which a model with the essential dynamics of a system is wanted and where this model should have a finite (and limited) number of parameters. Applications for such models include the simulation of the process, the prediction of future behaviour, controller design. For such goals the applicability of non-parametric models is limited.

The previously introduced subspace models are examples of *parametric* models that depend only on a limited number of parameters. Once the order of the system is selected, the model structure is defined and the parameters can be estimated. The obtained models can be good enough for the intended purpose. As the model structure is based on state space equations, these models are well suited for MIMO systems. For SISO systems, transfer functions provide usually a more compact representation of a system. Although the subspace models for SISO systems can be easily transformed to a transfer function, the result may be "suboptimal". That means that another model structure may give better results. In this chapter the so-called *Prediction-Error identification Methods* (PEM) will be discussed that estimate the parameters of models directly in a transfer function format.

Starting point for the PEM-models is that we assume that the unknown system G_0 can be represented by a linear time invariant system of a finite order (LTIFD system). The discrete time input signal $u(t)$ and output signal $y(t)$ are measured. The output $y(t)$ contains the response of the system on the input signal, but in addition there is a contribution from an unmeasurable disturbance $v(t)$, so

$$y(t) = G_0(z)\, u(t) + v(t), \tag{6.1}$$

where the LTIFD system G_0 is written as a transfer function $G_0(z)$. In general, this disturbance $v(t)$ can originate from several sources like measurement noise, effects of non-measured inputs, process disturbances and non-linearities. For the rest of this chapter, the character of this disturbance signal will be restricted to a certain *noise model*. It will be assumed that $v(t)$ is a signal with power spectrum $\Phi_v(\omega)$ that can be written as

$$v(t) = H_0(z)\, e(t), \tag{6.2}$$

in which $e(t)$ is a white noise with variance σ_e^2 and $H_0(z)$ is a stable, monic and minimum phase transfer function. As $H_0(z)$ is minimum phase it should not have zeros outside the unit circle and a stable inverse $H_0^{-1}(z)$ exists. The model $H_0(z)$ is a monic transfer function which means that the limit $H_0(\infty) = 1$. Basically this is a way to normalise the nominator and denominator, as any transfer function of which $H_0(\infty)$ has a nonzero limit can be changed into a monic transfer function by multiplying the nominator with the appropriate factor. For the parameter estimation later in this chapter the monic character of $H_0(z)$ avoids a possible ambiguity to set the variance of the noise signal $v(t)$. Starting from a given noise model $H_0(z)$ and noise variance σ_e^2, the same disturbance $v(t)$ can be obtained by increasing the gain of the model $H_0(z)$ and decreasing the variance σ_e^2 at the same time. By requiring that $H_0(z)$ is monic, the gain of the model can not be chosen arbitrary and only one combination of $H_0(z)$ and σ_e^2 remains to define the observed disturbance $v(t)$.

One more property of a monic transfer function becomes clear when it is expressed by means of an impulse response $h(k)$ ($k = 0, 1, 2, ...$), so

$$H_0(z) = \sum_{k=0}^{\infty} h(k)\, z^{-k} \tag{6.3}$$

To assure that the limit $H_0(\infty) = 1$, the first sample of the impulse response has to satisfy

$$h(0) = 1, \tag{6.4}$$

so

$$H_0(z) = 1 + \sum_{k=1}^{\infty} h(k)\, z^{-k}. \tag{6.5}$$

To conclude this overview of properties of $H_0(z)$, it is pointed out that the inverse of a monic transfer function is also monic.

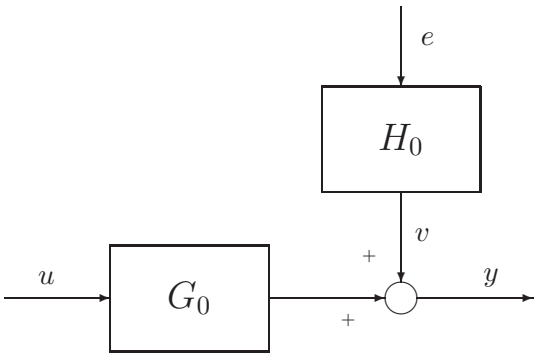


Figure 6.1: Schematic representation of the assumed system structure.

Combining Eqs. (6.1) and (6.2) gives a single equation for the assumed structure of the system

$$y(t) = G_0(z)\, u(t) + H_0(z)\, e(t). \tag{6.6}$$

Figure 6.1 gives a schematic representation of this structure. It has to be stressed that it is certainly not guaranteed that a real system fits into this structure. It is essential that the disturbance $v(t)$ can be generated from a white noise source $e(t)$. That implies that the disturbance $v(t)$ and input signal $u(t)$ can not be correlated. If e.g. $v(t)$ is used to account for non-linear behaviour of system G_0, this assumption is violated as this would mean that some (non-linear) relation exists between input $u(t)$ and (a part of) disturbance $v(t)$. Also the presence of a feedback controller that uses the output $y(t)$ to generate the input $u(t)$ of the system will introduce a relation between the disturbance $v(t)$ and the input signal $u(t)$. So in general it has to be verified that the *assumed* model structure is indeed valid for a given system.

6.2 Prediction and prediction error

As the name of the PEM-models suggest, the *prediction error* of the model plays an important role. This prediction error is the difference between the output $y(t)$ of the system as would be predicted by the model and the actual measurement. So first we need to know the *prediction* of our model.

The considered model is the general PEM-equation (6.6). In this section the subscript 0 will not be used for readability. To avoid another complication in the derived equations, it will be assumed that there is at least one sample time delay in $G(z)$, so $y(t)$ does not depend on $u(t)$ at the same time instant. The analysis below can be modified quite easily to deal with systems without delay. The question to be answered is:

What is the best prediction for the output $y(t)$ of the system given by Eq. (6.6) and given all past observations of input $u(s)$ and output $y(s)$ ($s \leq t-1$)?

For these past observations shorthand notations will be used:

$$\begin{aligned} U_{t-1} &= \{u(s), s \leq t-1\}, \\ Y_{t-1} &= \{y(s), s \leq t-1\}. \end{aligned} \tag{6.7}$$

To find the "best" prediction, first a simplified system will be considered in which there is no input $u(t)$, so

$$y(t) = v(t) = H(z)\, e(t). \tag{6.8}$$

All past outputs Y_{t-1} are known, which are equal to all past realisations of the disturbance V_{t-1} (using the same notation for $v(t)$ as introduced in Eq. (6.7)). To analyse this equation, we will make use of the properties of $H(z)$. Using the expression of Eq. (6.5) for $H(z)$, the next output equals

$$y(t) = v(t) = e(t) + \sum_{k=1}^{\infty} h(k)\, e(t-k). \tag{6.9}$$

This equation specifies how $v(t)$ is computed from $e(t)$ using $H(z)$. As $H(z)$ has a stable inverse, the reverse equation can also be evaluated, so

$$e(t) = H^{-1}(z)v(t). \tag{6.10}$$

As the inverse $H^{-1}(z)$ is also a monic transfer function, it can be written analogously to Eq. (6.5) as

$$H^{-1}(z) = 1 + \sum_{k=1}^{\infty} \bar{h}(k)\, z^{-k}, \tag{6.11}$$

where $\bar{h}(k)$ are the samples of the impulse response of $H^{-1}(z)$. When the model $H(z)$ is known, the inverse can be found and the samples of the impulse responses $h(k)$ and $\bar{h}(k)$ are also known. Then knowledge of Y_{t-1} implies that the past values of the white noise signal E_{t-1} can also be constructed. Consider e.g.

$$e(t-1) = v(t-1) + \sum_{k=1}^{\infty} \bar{h}(k)\, v(t-1-k), \tag{6.12}$$

and similar expressions hold for all other past values of $e(t)$. Combining Eqs. (6.9) and (6.12) gives

$$y(t) = v(t) = e(t) + m(t-1), \tag{6.13}$$

where the term $m(t-1)$ is completely known from past observations V_{t-1} and the impulse responses $h(k)$ and $\bar{h}(k)$ according to

$$m(t-1) = \sum_{k=1}^{\infty} h(k) \left(v(t-k) + \sum_{l=1}^{\infty} \bar{h}(l)\, v(t-k-l) \right). \tag{6.14}$$

From Eq. (6.13) the "best" prediction for output $y(t)$ can be easily computed. The expectation for the white noise signal is zero and $m(t-1)$ is completely known. The "best" prediction for $y(t)$ given all data up to $t-1$ is written as $\hat{y}(t|t-1)$ and thus equals

$$\hat{y}(t|t-1) = \hat{v}(t|t-1) := E\{v(t)|V_{t-1}\} = m(t-1), \tag{6.15}$$

in which the expression for $m(t-1)$ of Eq. (6.14) can be substituted to compute the result. A more compact expression for $m(t-1)$ is obtained by recognising that instead of Eq. (6.8) the disturbance v can also be written as

$$v(t) = e(t) + [H(z) - 1]\, e(t), \tag{6.16}$$

and the inverse relation (6.10) is rewritten as

$$e(t) = v(t) + [H^{-1}(z) - 1]\, v(t). \tag{6.17}$$

Combining both equations gives

$$v(t) = e(t) + [H(z) - 1]\left(v(t) + [H^{-1}(z) - 1]\, v(t)\right). \tag{6.18}$$

Expanding the right side of this equation gives after elimination of terms with opposite signs

$$v(t) = e(t) + [1 - H^{-1}(z)]\, v(t). \tag{6.19}$$

Comparing the second terms in Eqs. (6.13) and (6.19) shows that this is another way to write $m(t-1)$. Using this result the prediction in Eq. (6.15) can be written as

$$\hat{y}(t|t-1) = \hat{v}(t|t-1) = [1 - H^{-1}(z)]\, v(t). \tag{6.20}$$

At first sight it may look as if this expression uses not only past observations V_{t-1}, but also $v(t)$. This is not the case due to the monic character of $H^{-1}(z)$. As is clear from Eq. (6.11), the first term of the impulse response of $H^{-1}(z)$ cancels with the 1 in Eq. (6.20), so the first term with v is $v(t-1)$. Hence Eq. (6.20) is indeed a prediction for the output $y(t)$ based on past observations V_{t-1} (or Y_{t-1}).

Next the more general situation with $u \neq 0$ will be considered. Then $y(t)$ is given e.g. by Eq. (6.1) and U_{t-1} en Y_{t-1} are known. The past disturbances V_{t-1} are also known as all past values of $v(t)$ can be computed from

$$v(t) = y(t) - G(z)u(t), \tag{6.21}$$

where the righthand side is completely known up to and including time instant $t-1$. For the current time step, the best estimate for the output $y(t)$ consists of a deterministic part originating from the input $u(t)$ and an estimate for the next disturbance, so

$$E\{y(t)|U_{t-1}, Y_{t-1}\} = G(z)u(t) + E\{v(t)|V_{t-1}\}. \tag{6.22}$$

Then the determination of the best estimate of the output is again related to finding the best estimate for the disturbance at $v(t)$. This problem is identical to the situation for $u = 0$ as described above. When the disturbance v is known, the past values of the white noise source can be reconstructed. And with this E_{t-1} the best estimator for $v(t)$ is e.g. given by Eq. (6.19). So the prediction for the output becomes

$$\hat{y}(t|t-1) = G(z)u(t) + [1 - H^{-1}(z)]\, v(t). \tag{6.23}$$

With Eq. (6.21) this prediction can be written as

$$\hat{y}(t|t-1) = H^{-1}(z)G(z)u(t) + [1 - H^{-1}(z)]\, y(t). \tag{6.24}$$

Note again that the last term does not include the output $y(t)$ at the current time step as the term 1 in $1 - H^{-1}(z)$ is cancelled by an identical term that is found in $H^{-1}(z)$ due to the monic character, see Eq. (6.11).

With the prediction of Eq. (6.24), the prediction error $\varepsilon(t)$ can be written as

$$\varepsilon(t) := y(t) - \hat{y}(t|t-1) = H^{-1}(z)[y(t) - G(z)u(t)] \tag{6.25}$$

The meaning of this equation is illustrated by the lower part of Fig. 6.2. It shows that the system model $G(z)$, the measured input $u(t)$ and the measured output $y(t)$ are used to estimate the disturbance $\hat{v}$. With the inverse noise model $H^{-1}(z)$ the prediction error $\varepsilon(t)$ is finally computed.

Figure 6.2 shows more than a schematic representation of the calculation of the prediction error. It will also serve as a general scheme for the identification as will be discussed in the remainder of this chapter. As indicated in the top part of the figure, we will *assume* that our real system somehow fits in

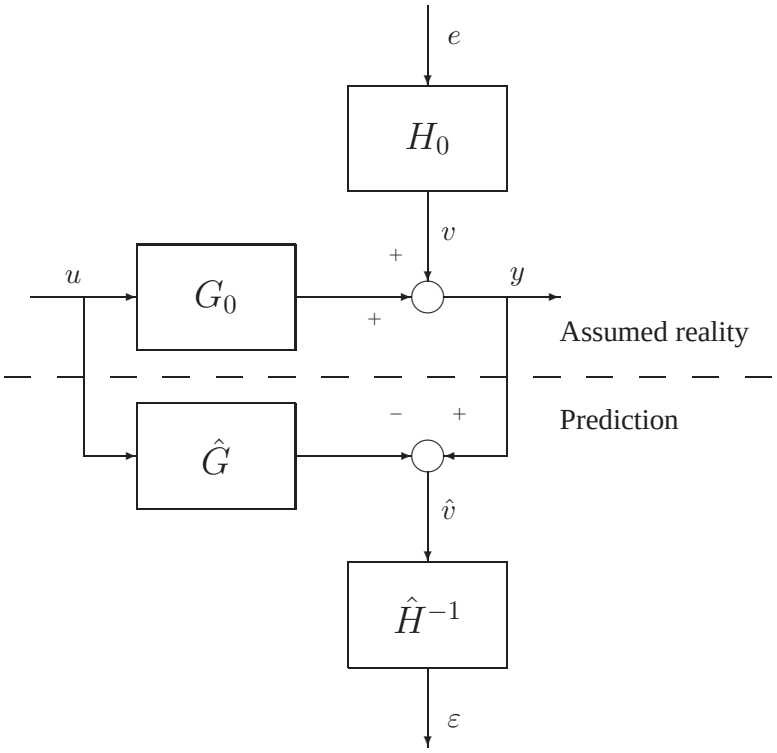


Figure 6.2: Schematic representation of the prediction error.

the PEM-framework of Eq. (6.6). So it is assumed that the data from the real system can be considered to be generated by a process model $G_0(z)$ and a noise model $H_0(z)$ as expressed by this equation. Then the goal of (system) identification in this chapter is to determine transfer functions $G(z)$ and $H(z)$ from the measured input and output data.

For that purpose, these transfer functions will be written as functions of parameters from a parameter vector θ. Next by varying the parameters the transfer functions $G(z)$ and $H(z)$ will be "tuned" to match the assumed process models $G_0(z)$ and $H_0(z)$. In Fig. 6.2, it can be seen that when the match is perfect, so the identified $G(z)$ and $H(z)$ are exactly equal to the "real" $G_0(z)$ en $H_0(z)$, the prediction error $\varepsilon(t)$ according to Eq. (6.25) is a white noise signal. In practice, this "tuning" is often carried out by minimising the error $\varepsilon(t)$ with a least squares fit as will be outlined later in this chapter.

6.3 Model structures

Before the models $G(z)$ and $H(z)$ can be identified, we first have to define how these models depend on the parameters θ and how many parameters are actually being used. Examples of this so-called parameterisations are:

- The coefficients of series like in impulse response models.
- Coefficients in state space matrices like in subspace identification.
- Coefficients in fractions of polynomials.

The latter parameterisations is used for PEM-models, so $G(z)$ and $H(z)$ are both written as rational functions. Coefficients of the powers of q in these polynomials are parameters of the model. Once a specific parameterisation has been defined, it implies that during the identification model candidates are considered from the *model set* $\mathcal{M}$ that is formally defined as

$$\mathcal{M} = \{(G(z,\theta), H(z,\theta)) \,|\, \theta \in \Theta \subset \mathbb{R}^d\}. \tag{6.26}$$

This expression indicates that the model set defines both models $G(z)$ and $H(z)$ and this combination depends on d real valued parameters θ. When each parameter can have an arbitrary value, the parameter vector spans the full d-dimensional space $\mathbb{R}^d$. When the values of one or more parameters are confined, the parameters θ must be chosen from a parameter range Θ that is a subspace of $\mathbb{R}^d$.

What remains is to define how the models depend exactly on the parameters. As a first example of such a parameterisation, we consider the so-called ARX model structure. In this model structure the rational functions are defined as

$$G(z,\theta) = \frac{B(z^{-1},\theta)}{A(z^{-1},\theta)} \quad \text{and} \quad H(z,\theta) = \frac{1}{A(z^{-1},\theta)}, \tag{6.27}$$

with the polynomials A and B

$$\begin{aligned} B(z^{-1},\theta) &= b_1 + b_2 z^{-1} + ... + b_{n_b} z^{-(n_b-1)} \quad \text{and} \\ A(z^{-1},\theta) &= 1 + a_1 z^{-1} + ... + a_{n_a} z^{-n_a}. \end{aligned} \tag{6.28}$$

The n_b coefficients of the B polynomial and the n_a coefficients of the A polynomial are collected in the parameter vector

$$\theta = [a_1\, a_2\, ...\, a_{n_a}\, b_1\, b_2\, ...\, n_{n_b}]^T \in \mathbb{R}^{n_a+n_b}. \tag{6.29}$$

In Eq. (6.28) it can be seen that there is a subtle difference between both polynomials, as the A polynomial is defined to start with a fixed coefficient 1. There are several reasons for this choice. First, if all coefficients in A and B are parameters, then an ambiguity arises from the definition of the transfer function $G(z)$. For any given set of parameter values exactly the same model would be obtained when all parameters are multiplied with the same nonzero constant. Apparently, no unique solution for the best model fit will exist. This ambiguity can be resolved by setting one coefficient to a fixed value. Setting the first coefficient of A equal to 1 has two advantages. The first advantage is that the monic property of $H(z)$ in Eq. (6.27) is also satisfied. The second reason for this choice is that discrete time transfer functions are often normalised this way as it shows how the output $y(t)$ can actually be computed. That becomes clear when Eq. (6.27) is substituted in the PEM structure like in Eq. (6.6)

$$y(t) = \frac{B(z^{-1},\theta)}{A(z^{-1},\theta)}\, u(t) + \frac{1}{A(z^{-1},\theta)}\, e(t). \tag{6.30}$$

This can be rewritten as

$$A(z^{-1},\theta)\, y(t) = B(z^{-1},\theta)\, u(t) + e(t). \tag{6.31}$$

Substituting Eq. (6.28) in this result gives

$$(1 + a_1 z^{-1} + ... + a_{n_a} z^{-n_a})\, y(t) = (b_1 + b_2 z^{-1} + ... + b_{n_b} z^{-(n_b-1)})\, u(t) + e(t), \tag{6.32}$$

and

$$\begin{aligned} &y(t) + a_1 y(t-1) + ... + a_{n_a} y(t-n_a) \\ &\qquad = b_1 u(t) + b_2 u(t-1) + ... + b_{n_b} u(t-(n_b-1)) + e(t). \end{aligned} \tag{6.33}$$

From the latter equation is follows quite straightforward how $y(t)$ can be computed from past outputs, inputs and the current white noise realisation, namely

$$\begin{aligned} y(t) = &-a_1 y(t-1) - ... - a_{n_a} y(t-n_a) \\ &+b_1 u(t) + b_2 u(t-1) + ... + b_{n_b} u(t-(n_b-1)) + e(t). \end{aligned} \tag{6.34}$$

If the first coefficient in the A polynomial is unequal 1, then this coefficient will show up as the coefficient of $y(t)$ in the lefthand side of this equation, so the righthand side expression would have to

be divided by this factor to compute $y(t)$. This would involve extra checks to assure that the factor is nonzero. Apparently, taking this term equal to 1 is a good choice.

The name "ARX" for these type op models can be explained from the terms in Eq. (6.31). "AR" means AutoRegressive and refers to the part of the output y that is generated from the white noise source e. The "X" refers to the eXogenous term with the u that indicates that another contribution to the output is the response to an input signal.

As these ARX models are examples of PEM-models, the theory of predictions and prediction errors from Eqs. (6.24) and (6.25) respectively is applicable. Using the expressions for $G(z)$ and $H(z)$ from Eq. (6.27), the prediction is

$$\hat{y}(t|t-1;\theta) = B(z^{-1},\theta)\,u(t) + [1 - A(z^{-1},\theta)]\,y(t), \tag{6.35}$$

where explicitly the dependence on the parameter vector θ is indicated. Note that $G(z)$ does not have a delay from input to output and for that reason the current input $u(t)$ is needed to estimate the current output $y(t)$.

It appears that the expression for the prediction is linear in any of the parameters from θ. The same is true for the prediction error that can be written as

$$\begin{aligned} \varepsilon(t,\theta) &= y(t) - \hat{y}(t|t-1;\theta) \\ &= y(t) + a_1 y(t-1) + ... + a_{n_a} y(t-n_a) \\ &\qquad -b_1 u(t) - b_2 u(t-1) - ... - b_{n_b} u(t-(n_b-1)). \end{aligned} \tag{6.36}$$

One way to identify the system is to find the parameter set $\hat{\theta}$ that minimises $\varepsilon(t,\theta)$ in a least squares sense. As the prediction error is linear in the parameters, this can be formulated as a linear least squares problem. To obtain the solution, an output vector is constructed from the measured output like in

$$Y_N = \begin{bmatrix} \vdots \\ y(t) \\ \vdots \end{bmatrix}, \tag{6.37}$$

and a regression matrix is build from the past outputs and inputs

$$\Phi_N = \begin{bmatrix} \vdots & & \vdots & \vdots & & \vdots \\ -y(t-1) & ... & -y(t-n_a) & u(t) & ... & u(t-(n_b-1)) \\ \vdots & & \vdots & \vdots & & \vdots \end{bmatrix}. \tag{6.38}$$

With these expression normal equations can be derived and the parameter estimation is

$$\hat{\theta} = \Phi_N^{\dagger} Y_N. \tag{6.39}$$

In this expression the pseudo-inverse of Φ_N is used, but for a numerical solution other algorithms are more favourable, i.e. MATLAB's matrix left divide `mldivide` (Sect.4.1.3). In any case it appears that identification with ARX models can lead to a linear least squares problem for which the unique and global optimum parameters set can be found reliably and quickly. That is an advantage of ARX models compared to some other PEM model structures as will be outlined next.

Analogously to the ARX model structure, other model structures for PEM models can be defined. Essentially they differ in the way the rational fractions for $G(z)$ and $H(z)$ are expressed in polynomials. Table 6.1 gives an overview of the most frequently used PEM models.

ARX The ARX model structure has been introduced in the previous section. It is characterised by the common denominator $A(z^{-1},\theta)$ for system and noise model. The nominator of the noise model is fixed to a constant 1. It may be questioned whether a real model fits into this structure, but the large advantage of this structure is the linearity in the parameters.

Name	equation	$G(z,\theta)$	$H(z,\theta)$
ARX	$A(z^{-1})\,y(t) = B(z^{-1})\,u(t) + e(t)$	$\frac{B(z^{-1},\theta)}{A(z^{-1},\theta)}$	$\frac{1}{A(z^{-1},\theta)}$
ARMAX	$A(z^{-1})\,y(t) = B(z^{-1})\,u(t) + C(z^{-1})\,e(t)$	$\frac{B(z^{-1},\theta)}{A(z^{-1},\theta)}$	$\frac{C(z^{-1},\theta)}{A(z^{-1},\theta)}$
OE	$y(t) = \frac{B(z^{-1})}{F(z^{-1})}\,u(t) + e(t)$	$\frac{B(z^{-1},\theta)}{F(z^{-1},\theta)}$	1
FIR	$y(t) = B(z^{-1})\,u(t) + e(t)$	$B(z^{-1},\theta)$	1
BJ	$y(t) = \frac{B(z^{-1})}{F(z^{-1})}\,u(t) + \frac{C(z^{-1})}{D(z^{-1})}\,e(t)$	$\frac{B(z^{-1},\theta)}{F(z^{-1},\theta)}$	$\frac{C(z^{-1},\theta)}{D(z^{-1},\theta)}$

Table 6.1: Overview of PEM model structures.

ARMAX In an ARMAX model the "AR" structure with the common denominator $A(z^{-1},\theta)$ and the "X" contribution of an external input are also present. The "MA" refers to a Moving Average that is introduced by an extra $C(z^{-1},\theta)$ polynomial in the nominator of the noise model. This polynomial has to start with a constant term equal 1 to assure the monic character of $H(z)$.

OE The Output Error model structure assumes that the disturbance is a white noise signal, so $H(z) = 1$. For this model structure the denominator of $G(z)$ is described with a polynomial $F(z^{-1},\theta)$ instead of the $A(z^{-1},\theta)$. This way the $A(z^{-1},\theta)$ is reserved for the *common* part in the denominators of system and noise models.

FIR The Finite Impulse Response only has a $B(z^{-1},\theta)$ polynomial with the impulse response of the system. In principle this expression is identical to Eq. (3.4) for the discussion of non-parametric identification, except that it is now written in the structure of a PEM-model. And of course PEM-techniques are used to find the parameters instead of the correlation analysis of chapter 3. Finally note that a FIR-model is a special case of an ARX model having $A = 1$ (so $n_a = 0$).

BJ The Box-Jenkin model structure provides the most general structure. Both system and noise are modelled by polynomial fractions and there are no assumptions about common parts in the denominators.

In the `ident` toolbox the model structures `arx`, `armax`, `oe` and `bj` are defined according to the structures defined in Table 6.1. In order to complete the definition of a specific model set $\mathcal{M}$ the model structure has to be chosen and the number of parameters for each polynomial has to be set. Then a model can be estimated by estimating the parameters. In `ident` the obtained models are given a name that reflects both the model structure and the number of parameters. These numbers are given in alphabetic order of the polynomials that are used.

A typical ARX model could be `arx231` which means that $n_a = 2$ and $n_b = 3$. The final number is denoted n_k, so $n_k = 1$, and refers to a number of extra delays that are added to the system model of any PEM model. That means that the actual model equation becomes

$$y(t) = G(z,\theta)z^{-n_k}\,u(t) + H(z,\theta)\,e(t). \tag{6.40}$$

This way it appears as if the coefficients in the nominator of the system shift n_k terms. For example in an ARX model we get

$$G(z,\theta)z^{-n_k} = \frac{b_1 z^{-n_k} + b_2 z^{-(n_k+1)} + ... + b_{n_b} z^{-(n_k+n_b-1)}}{1 + a_1 z^{-1} + ... + a_{n_a} z^{-n_a}}. \tag{6.41}$$

Note that an alternative approach to account for n_k delays is to enlarge the $B(z^{-1},\theta)$ polynomial with n_k more terms (and n_k more parameters) and setting the first n_k coefficients of this polynomial to a fixed value equal to zero. As that would complicate the expressions for the parameter estimation, defining an explicit delay as in Eq. (6.40) is more elegant.

For ARMAX models the name of a model includes four numbers for n_a, n_b, n_c and n_k respectively. For OE and BJ models it should be noted that n_f is listed *after* n_b, so `oe321` refers to a model that has a similar structure for the system model as `arx231`.

Properties of the model structures

In the description of the ARX model structure it was already pointed out that this model structure results in a prediction and prediction error that is linear in the parameters. That is clearly an advantage as it means that e.g. linear least squares techniques can be applied to obtain the parameter estimate $\hat{\theta}$. From Eq. (6.24) it can be concluded that this is the case if both $H^{-1}(z)G(z)$ and $[1-H^{-1}(z)]$ are polynomials. The model structures that satisfy this criterion are ARX and FIR, so only for these model structures a fast algorithm is available. Therefore these model structures are often used to estimate the order of a system. The parameter estimates for model structures ARMAX, OE and BJ that are nonlinear in the parameters have to be found with nonlinear optimisation techniques.

One other property of the model structures is the independent parameterisation of $G(z,\theta)$ and $H(z,\theta)$. That means that no common parameters exist in G and H and an independent identification of these transfer functions will be possible. For the structures in Table 6.1 this is the case when there is no A polynomial, so for OE, FIR and BJ. In ARX and ARMAX models the denominators of system and noise model are identical and there is no independent parameterisation of $G(z,\theta)$ and $H(z,\theta)$.

6.4 Identification criterion

Identifying a model from measured data requires the following ingredients:

- the experimental data $\{(y(t),u(t)), t = 1,...,N\}$.
- the model set $\mathcal{M}$ which is defined by choosing a model structure and the number of parameters that need to be identified. Setting the number of delays n_k in Eq. (6.40) is also part of this definition.
- the identification criterion to be discussed below.

When these three ingredients are available, the rest is essentially solving the mathematical (and numerical) problem of finding the parameters set that satisfies the identification criterion for the given data and model set.

For PEM methods the identification criterion is defined by considering the prediction error $\varepsilon(t,\theta)$ as in Eq. (6.25) for all sample times t as a function of the parameters θ. As was concluded from Fig. 6.2 a perfect match between identified model and (assumed) reality will result in a prediction error that is a white noise signal. The identification criterion has to be chosen such that a good model fit is obtained. In practice the following criterions are mostly used:

- Least squares criterion: The minimisation of a scalar function of $\varepsilon(t,\theta)$. This cost function is computed either directly from the prediction error

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N} \varepsilon(t,\theta)^2 \tag{6.42}$$

or from a filtered error according to

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}(L(z)\varepsilon(t,\theta))^2. \tag{6.43}$$

- Alternatively, the so-called Instrumental Variable (IV) techniques try to assure that the prediction error $\varepsilon(t,\theta)$ is indeed an uncorrelated signal. This will be discussed later.

For an identified model, the consistency is an important property. For the least squares criterion consistency is guaranteed in the following cases (for a proof see e.g. Ljung [6, Chapter 8]):

1. If the (real) system (G_0, H_0) is in the chosen model set and the power spectrum of the input signal $\Phi_u(\omega)$ is nonzero in sufficient frequencies (i.e. the input signal is *sufficiently exciting*), then $G(z,\theta_N)$ en $H(z,\theta_N)$ are consistent estimators.

2. If the system G_0 is in the chosen model set, G and H are parameterised independently (e.g. FIR, OE and BJ models) and the power spectrum of the input signal $\Phi_u(\omega)$ is nonzero in sufficient frequencies, then $G(z,\theta_N)$ is a consist estimator.

In practice, it is not so difficult to create an input signal that is sufficiently exciting. The main difficulty in applying one of these guarantees for consistency, is that the real system has to be in the model set used for identification. Especially the first case states that only *both* models can be estimated consistently or you have no guarantee at all. So even if one is only interested in the system model G, then still the noise model has to chosen carefully. This is relaxed by the second case where only the real system has to be in the model set, but then an independent parameterisation for G and H should be used, i.e. no ARX of ARMAX models.

Example of the consistency properties

The consistency of the PEM models is illustrated with the second order model system of appendix A.1:

$$y_k - 1.5y_{k-1} + 0.7y_{k-2} = u_{k-1} + 0.5u_{k-2} + e_k - e_{k-1} + 0.2e_{k-2}. \tag{6.44}$$

It can be rewritten as a PEM model with

$$G_0(z) = \frac{z^{-1} + 0.5z^{-2}}{1.0 - 1.5z^{-1} + 0.7z^{-2}} \quad \text{and} \quad H_0(z) = \frac{1.0 - 1.0z^{-1} + 0.2z^{-2}}{1.0 - 1.5z^{-1} + 0.7z^{-2}} \tag{6.45}$$

This corresponds to an ARMAX model structure with

$$\begin{array}{rcl} A(z^{-1}) &=& 1.0 - 1.5z^{-1} + 0.7z^{-2} \\ B(z^{-1}) &=& z^{-1} + 0.5z^{-2} \\ C(z^{-1}) &=& 1.0 - 1.0z^{-1} + 0.2z^{-2} \end{array} \tag{6.46}$$

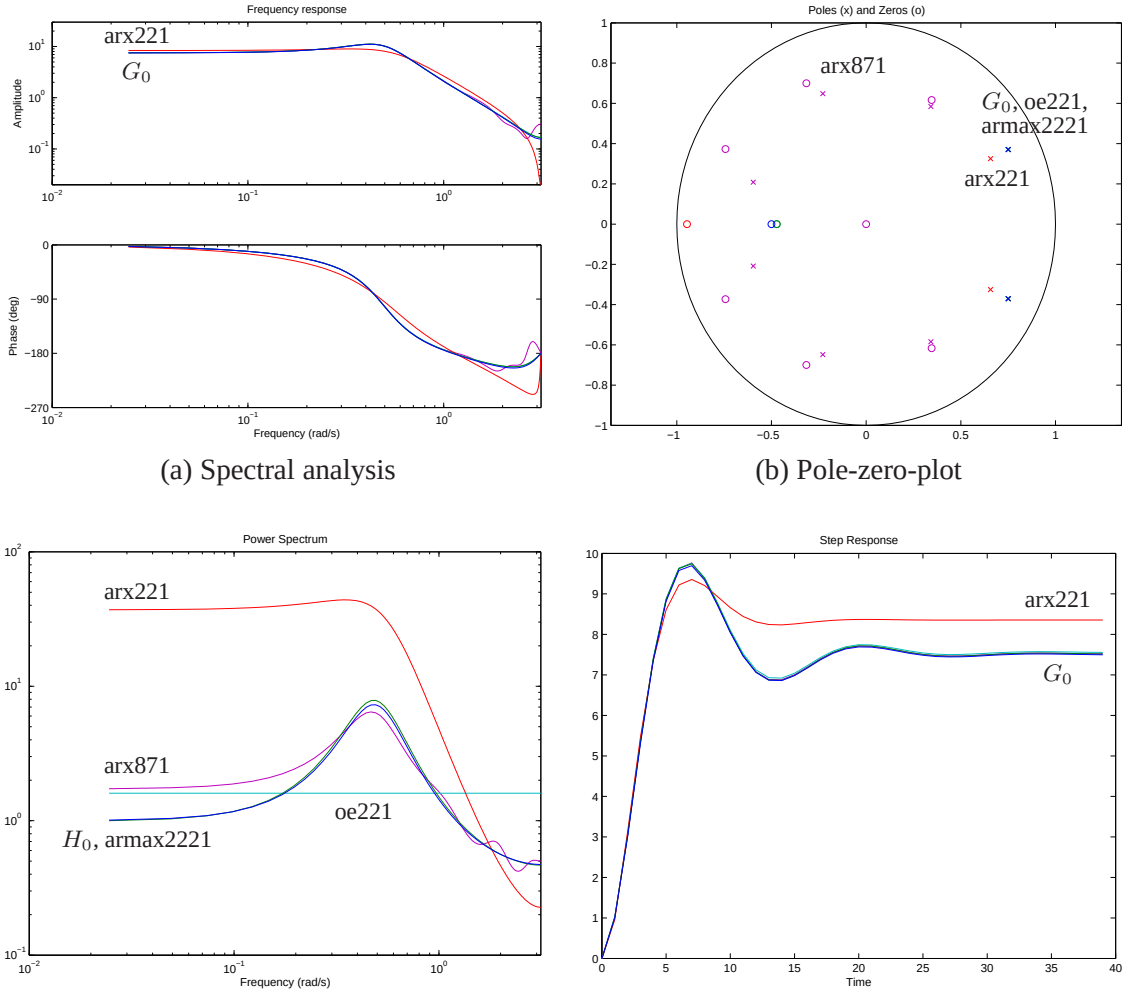
Input and output data for this system has been simulated as outlined in the appendix and the data is used to estimate PEM models. The following models will be considered:

G_0**,** H_0 The real system.

`armax2221` An ARMAX model with $n_a = 2$, $n_b = 2$, $n_c = 2$ and $n_k = 1$, Note that both G_0 and H_0 of the real systems fit into this model set.

`arx221` An ARX model with $n_a = 2$, $n_b = 2$ and $n_k = 1$. In this case only G_0 fits in the model set as the nominator of H_0 can not be represented in the ARX model.

`oe221` An OE model with $n_b = 2$, $n_f = 2$ and $n_k = 1$, so essentially the same system model as for the ARX model above. Also in this case only G_0 fits in the model set as the noise model of an OE model always equals 1 and can not represent a transfer function like H_0.



(a) Spectral analysis

(b) Pole-zero-plot

(c) Noise spectrum

(d) Step response

Figure 6.3: Analysis of several identified models from data of a second order model system.

`arx871` A high order ARX model ($n_a = 8$, $n_b = 7$ and $n_k = 1$).

Figure 6.3 shows an analysis of these identified models in comparison with the data generating systems G_0 and H_0. The spectral analysis, Fig. 6.3(a), is very similar for the `armax2221` model and G_0. Actually, the `armax2221` can not be distinguished from the real system G_0 indicating a very good model fit. From the consistency properties this is expected as the real systems (G_0 and H_0) fit perfectly in this model set and (apparently) the input is sufficiently exciting. A similar conclusion is obtained for the `oe221` model. In this case only G_0 fits in the model set, but consistency can still be obtained as $\hat{G}$ and $\hat{H}$ are parameterised independently.

That is not true for the `arx221` model. It has essentially the same (and correct) model structure for $\hat{G}$, but because there is no independent parameterisation there is no guarantee for a consistent fit as H_0 does not fit in the model set. The frequency response of this `arx221` model is indeed different from G_0, see e.g. the height of the peak near 0.4 rad/s. When it is still desirable to use an ARX model, it is often attempted to obtain a better model fit with a higher order model. According to the frequency response, this helps indeed as the `arx871` model agrees well with G_0 up to about 1 rad/s.

The pole-zero plot, Fig. 6.3(b), shows that the extra poles and zeros of the high order `arx871` somehow cluster into pairs, so the low frequent behaviour is dominated by the low frequency poles. These agree well with the results for the `armax2221` and `oe221` models and the real system G_0. Clearly, the poles and zero of the `arx221` model differ from these results indicating once more the lack of consistency of this model.

The noise spectrum, Fig. 6.3(c), shows most differences between the models. The `armax2221` model matches best with H_0, although there is a small (hardly visible) difference near 0.5 rad/s. The noise model of the `oe221` model has a constant spectrum as is clearly visible from the graph. The large difference between H_0 and the `arx221` model once more shows that the real noise model does not agree with the identified noise model. It can also be seen that the extra poles in the high order `arx871` model can indeed be used to obtain a better agreement for the noise model, although it can never be perfect. As these poles are not needed for the system model, they are to some extend cancelled by zeros as was already discussed from the pole-zero plot.

It should also be pointed out that a larger result for the noise spectrum $\Phi_v(\omega)$ can also be interpreted as larger model errors. This can be understood from the meaning of the disturbance v. It should account for the part of the output y that is not generated as the response of the system $\hat{G}$ to the input signal u. When the resemblance between the identified system $\hat{G}$ and the real system G_0 is bad, then according to Fig. 6.2 a larger $\hat{v}$ will result. This signal has to be modelled by $\hat{H}$ and the estimated variance of the white noise source. The power spectrum is computed from these results, so a larger power spectrum is likely to result. Then it is clear from Fig. 6.3(c) that the `arx221` model probably has the worst model fit.

Finally the step response, Fig. 6.3(d), illustrates also the difference between the `arx221` model and all other results. All consistent estimates for G_0 agree well. The `arx221` model gives still a good result for the first samples of the step response, but in particular the final value is wrong by about 10%.

To summarise, it can be concluded that a perfect identification can be carried out with the correct model set, see `armax2221`. With the `oe221` model still a consist estimate for the system model is found. However, the `arx221` does not give consistent estimates of the system and noise models. That implies that it is not possible to improve the quality of the fit e.g. by using more data. The variances of the estimated models will decrease, but the models will converge to the wrong answer. If for some reason an ARX model is desired, a higher order model with more parameters gives a reasonable agreement with the real system.

Asymptotic variance

For a system that fits in the model set, it can be proven [6, Chapter 9]) in the case the input and disturbance are uncorrelated, the number of parameters $n \to \infty$, the number of samples $N \to \infty$ while $n/N \to 0$ (so $n \ll N$), that the covariance of the system model is given by

$$\mathrm{var}(\hat{G}_N(e^{i\omega T})) \sim \frac{n}{N}\frac{\Phi_v(\omega)}{\Phi_u(\omega)}. \tag{6.47}$$

This expressions indicates that the estimate of the system model will be more accurate (in the frequency domain) if more measurements (N) are taken or if the input is more exciting ($\Phi_u(\omega)$). On the other hand, a larger number of parameters (n) or a large disturbance ($\Phi_v(\omega)$) decrease the accuracy. So one should take care of a sufficient signal to noise ratio and the number of measurements have to scale with the number of parameters.

A similar relation for the noise model

$$\mathrm{var}(\hat{H}_N(e^{i\omega T})) \sim \frac{n}{N}\frac{\Phi_v(\omega)}{\sigma_e^2} = \frac{n}{N}|H_0(e^{i\omega})|^2. \tag{6.48}$$

shows that the accuracy of this model (in the frequency domain) can be influenced by the ratio of the number of parameters (n) and the number of measurements (N).

6.5 Approximate modelling

In many practical cases the consistency requirements are not realistic. Then the question arises what is the outcome of the identification if the real system does *not* fit in the model set.

This question can be answered in the time domain and the frequency domain. Although we are dealing with time domain data, it will appear that a frequency domain analysis gives a lot of insight. The largest part of this section will deal with this frequency domain analysis. At the end some time domain related remarks are given. The expressions in the frequency domain are simplified by taking the sample time $T = 1$.

We recall that PEM models focus on the prediction error that equals

$$\varepsilon(t,\theta) = H^{-1}(z,\theta)[y(t) - G(z,\theta)u(t)]. \tag{6.49}$$

Assuming that the data generating system can be described with systems G_0 and H_0 as in Fig. 6.2, the prediction error can also be written as

$$\varepsilon(t,\theta) = H^{-1}(z,\theta)[(G_0(z) - G(z,\theta))u(t) + v(t)], \tag{6.50}$$

where the disturbance is assumed to be generated as the output of system H_0 with a white noise source of variance σ_e as input signal. Then input u and the disturbance v are uncorrelated and the power spectrum is

$$\Phi_\varepsilon(\omega,\theta) = \frac{|G_0(e^{i\omega}) - G(e^{i\omega},\theta)|^2\,\Phi_u(\omega) + \Phi_v(\omega)}{|H(e^{i\omega},\theta)|^2}, \tag{6.51}$$

with

$$\Phi_v(\omega) = \sigma_e^2 |H_0(e^{i\omega})|^2. \tag{6.52}$$

The parameters are found by minimisation of the cost function

$$V_N = \frac{1}{N}\sum_{t=1}^{N} \varepsilon(t,\theta)^2. \tag{6.53}$$

In the case $N \to \infty$ the limit of the cost function is the expectation according to

$$V_N \to \bar{V}(\theta) = \bar{E}\varepsilon(t,\theta)^2. \tag{6.54}$$

This time domain expression can be transformed to the frequency domain using Parseval's relationship. The result is

$$\bar{V}(\theta) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \Phi_\varepsilon(\omega,\theta)\,\mathrm{d}\omega \tag{6.55}$$

and the limit θ^* to which the estimator θ_N converges for $N \to \infty$ should minimise this expression. Substitution of the power spectrum Eq. (6.51) gives

$$\bar{V}(\theta) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \frac{|G_0(e^{i\omega}) - G(e^{i\omega},\theta)|^2\,\Phi_u(\omega) + \Phi_v(\omega)}{|H(e^{i\omega},\theta)|^2}\,\mathrm{d}\omega \tag{6.56}$$

that has to be minimised. Two mechanisms can be distinguished:

- Minimising $\dfrac{|G_0(e^{i\omega}) - G(e^{i\omega},\theta)|^2\,\Phi_u(\omega)}{|H(e^{i\omega},\theta)|^2}$.
- Fitting of nominator (with $\Phi_v(\omega)$ term) and denominator.

The first mechanism is to some extend desired as it means that $G_0(z)$ is indeed approximated by $G(z,\theta)$ in the frequency domain. However, the details of this fitting depend on the noise model $H(z,\theta)$, so in fact some a priori unknown weighting occurs. Furthermore, the presence of the noise power spectrum $\Phi_v(\omega)$ complicates the interpretation of Eq. (6.56).

Example: Fourth order model system

To illustrate the approximate modelling, the fourth order system of Appendix A.2 is considered. This system is simulated without a disturbance and the output is

$$y(t) = G_0(z)u(t) \tag{6.57}$$

with a fourth order system model

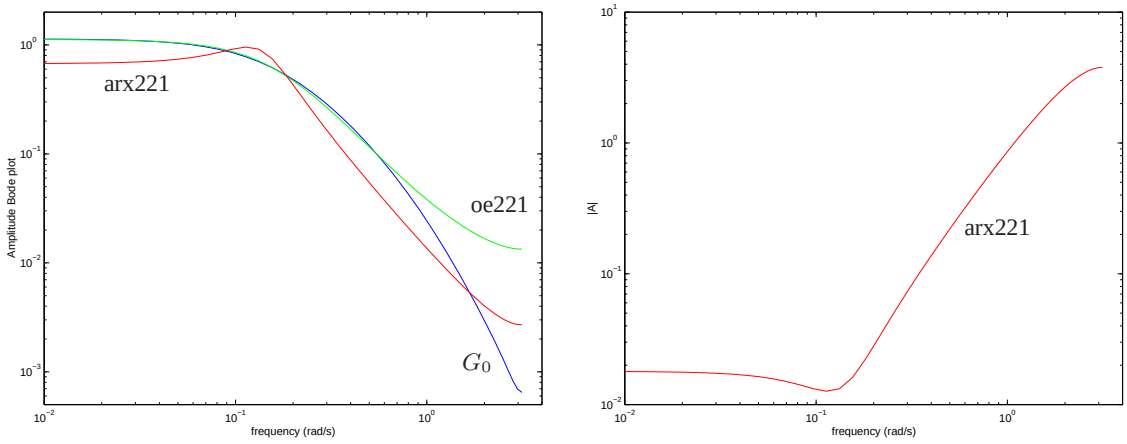
$$G_0(z) = \frac{0.001z^{-2}(10 + 7.4z^{-1} + 0.924z^{-2} + 0.1764z^{-3})}{1 - 2.14z^{-1} + 1.553z^{-2} - 0.4387z^{-3} + 0.042z^{-4}}. \tag{6.58}$$

Data is generated with a PRBS input signal $u(t)$ with sample time 1 s and variance 1, so $\Phi_u(\omega) \approx 1$.

Two approximate models are estimated:

- 2nd order OE (`oe221`): $y(t) = \dfrac{b_1\, z^{-1} + b_2\, z^{-2}}{1 + f_1\, z^{-1} + f_2\, z^{-2}}\, u(t) + e(t)$
- 2nd order ARX (`arx221`): $(1 + a_1\, z^{-1} + a_2\, z^{-2})\, y(t) = (b_1\, z^{-1} + b_2\, z^{-2})\, u(t) + e(t)$

Both models have identical system models $\hat{G}$, but differ in the noise models. Obviously, both models are too simple to capture the complete dynamic behaviour of G_0.



(a) Spectral analysis (b) Inverse ARX noise model

Figure 6.4: (a) Spectral analysis of two identified low order models from data of a fourth order model system. (b) Inverse noise model of the identified ARX model.

Figure 6.4(a) shows the spectral analyses of both identified models and the original G_0. Clearly, the low order models can not match the behaviour of fourth order system in the full frequency range. The `oe221` model shows a good agreement up to about 0.5 rad/s. The `arx221` model does not match well with G_0 in any frequency range, but to some extend the agreement with the real system in the high frequency range is improved compared to `oe221`.

This observation is typical for these kind of model structures: OE models tend to be better at low frequencies and ARX models have a better agreement at high frequencies. This difference can be understood from Eq. (6.56), as for OE models the noise model $\hat{H}$ is constant for all frequencies. As a result the weighting of the difference $|G_0(e^{i\omega}) - G(e^{i\omega}, \theta)|$ is also similar for all frequencies. For most systems $|G_0(e^{i\omega})|$ decreases at high frequencies, so the *relative* error in the estimated transfer function of an OE model will be larger at high frequencies. In contrast the spectrum of the ARX noise model depends on the A polynomial. Equation (6.56) shows that the fit between $G(z, \hat{\theta}_N)$ and $G_0(e^{i\omega})$ is weighted by the inverse (identified) noise model, so for an ARX model by

$$\frac{1}{|H(e^{i\omega}, \theta)|^2} = |A(e^{i\omega}, \theta)|^2. \tag{6.59}$$

As this noise model has only zeros and no poles, it is expected that will be larger for high frequencies. Figure 6.4(b) illustrates this for the identified `arx221` model. Clearly, the fit of the ARX model will emphasise model errors at high frequencies. If a too low order ARX model is used, then the available parameters will be used preferably to model the high frequency behaviour. A complication is that the weighting function Eq. (6.59) depends on the identified model, so it is not known beforehand.

Approximate modelling with a fixed noise model

One way to avoid the weighting with an unknown function, is to use a fixed noise model, so

$$H(e^{i\omega}, \theta) = H_*(z), \tag{6.60}$$

with some $H_*(z)$ that does not depend on any of the parameters θ. Equation (6.56) then becomes

$$\bar{V}(\theta) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \frac{|G_0(e^{i\omega}) - G(e^{i\omega}, \theta)|^2 \, \Phi_u(\omega) + \Phi_v(\omega)}{|H_*(e^{i\omega})|^2} \, \mathrm{d}\omega. \tag{6.61}$$

The mechanism that minimises this cost function is much easier to understand as the constant ratio $\Phi_v(\omega)/|H_*(e^{i\omega})|^2$ can be removed. What remains is the minimisation of

$$\frac{1}{2\pi} \int_{-\infty}^{\infty} |G_0(e^{i\omega}) - G(e^{i\omega}, \theta)|^2 \, \frac{\Phi_u(\omega)}{|H_*(e^{i\omega})|^2} \, \mathrm{d}\omega. \tag{6.62}$$

This expression shows that the least squares estimate $G(e^{i\omega}, \theta)$ of $G_0(e^{i\omega})$ is found by applying a frequency domain weighting function $\Phi_u(\omega)/|H_*(e^{i\omega})|^2$. The weighting can be modified by choosing the input spectrum $\Phi_u(\omega)$ and by the imposed noise model $H_*(e^{i\omega})$.

Obviously, for all OE-models Eq. (6.60) is true as $H_*(z) = 1$. For such models the weighting in the frequency domain still depends on the spectrum of the input signal $\Phi_u(\omega)$.

Approximate modelling with a prefilter $L(z)$

Another way to deal with the implicit weighting with the inverse noise model in the PEM models is to apply a prefilter $L(z)$ to both input and output data. It can be understood from Fig. 6.2 that inserting such a filter for both the measured u and y is equivalent to filtering ε with this filter. Consequently, the cost function to be minimised changes into a new cost function

$$V_N = \frac{1}{N} \sum_{t=1}^{N} \varepsilon_F(t, \theta)^2 = \frac{1}{N} \sum_{t=1}^{N} (L(z) \, \varepsilon(t, \theta))^2. \tag{6.63}$$

As before, we consider the asymptotic limit for $N \to \infty$

$$V_N \to \bar{V}_F(\theta) = \bar{E} \varepsilon_F(t, \theta)^2, \tag{6.64}$$

and once more with Parseval's relationship, the limit θ^* to which the estimator θ_N converges for $N \to \infty$ can be expressed in the frequency domain

$$\bar{V}_F(\theta) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \{|G_0(e^{i\omega}) - G(e^{i\omega}, \theta)|^2 \, \Phi_u(\omega) + \Phi_v(\omega)\} \, \frac{|L(e^{i\omega})|^2}{|H(e^{i\omega}, \theta)|^2} \, \mathrm{d}\omega. \tag{6.65}$$

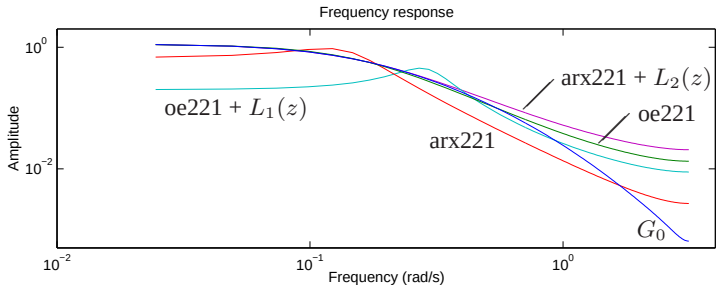
Compared to Eq. (6.56) the weighting (or filtering) of the approximation of G_0 by $G(z, \hat{\theta}_N)$ now depends on the modified noise model $L^{-1}(z)H(z, \theta)$, of which the prefilter $L(z)$ can be tuned by the user. As before, the approximation depends on the identified noise model $H(e^{i\omega}, \theta)$ and the interpretation of the minimisation is complicated by the presence of the $\Phi_v(\omega)$. However, in many cases the prefilter can be used to improve the model fit in the frequency range where a good performance is desired.

Example: Fourth order model system extended with prefilter for identification

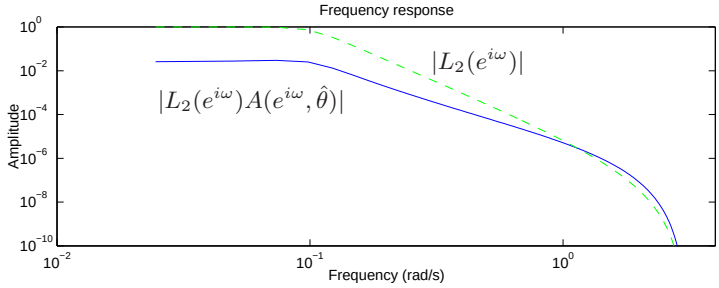
The same fourth order as defined before in Eq. (6.57) with the same date are used. ARX and OE models of too low order are identified. The previous results were obtained with no prefilter, so $L(z) = 1$.

The OE model does not follow the spectrum of the real system at frequencies above 0.5 rad/s. The quality of the fit can be improved by applying a high-pass prefilter $L_1(z)$, e.g. a fifth order high-pass Butterworth with a cut-off frequency of 0.5 rad/s. Figure 6.5(a) shows the spectrum of the previous models and the new OE-model. Clearly, the match with G_0 at higher frequencies is somewhat improved at the expense of a quality loss at low frequencies.

In practice, the aim is mostly not to improve the quality of an OE model at high frequencies, but to shift the focus of an ARX model to lower frequencies. With that goal a low-pass prefilter $L_2(z)$ should be applied, e.g. a fifth order low-pass Butterworth with a cut-off frequency of 0.1 rad/s. Figure 6.5(a) includes also the ARX model that is identified using this prefilter. It appears that practically the same low frequent behaviour is obtained as for the `oe221` model without prefilter.



(a) Spectral analysis



(b) Frequency weighting for the ARX model with prefilter

Figure 6.5: (a) Spectral analysis of two identified low order models from data of a fourth order model system with and without prefiltering. (b) Weighting of the prefilter $L_2(z)$ (dashed) and combined with the ARX noise model (solid).

The actual frequency weighting of e.g. an ARX model has to be interpreted with some care, as it depends not only on the prefilter, but also on the (a priori unknown) identified noise model. Figure 6.5(b) illustrates this for the identified `arx221` with prefilter $L_2(z)$. The dashed line is the spectrum of the chosen prefilter $L_2(z)$, which is a fifth order low-pass Butterworth filter. After the model has been estimated, the ultimate weighting filter $|L_2(e^{i\omega})A(e^{i\omega}, \hat{\theta})|$ can be computed from the A polynomial. This filter is the solid line in the graph. For an ARX model the inverse noise model is typically a high-pass filter, so the order of the ultimate weighting function will be less than the order of $L_2(z)$. The effect of a chosen prefilter may therefore be less than expected. The difference between the initial prefilter and the ultimate weighting depends on the order of the (inverse) noise model, so for an ARX model on the order of the A polynomial. When higher order models are estimated, it may be necessary to use prefilters of at least comparable order.

As was pointed out, the OE model without prefilter and the ARX model with low-pass prefilter are quite similar at low frequencies. In practice there may be reasons to prefer the ARX model(s) for

their computational advantages. ARX models with and without a prefilter can be computed uniquely and quickly from a linear optimisation problem, whereas the identification of OE models involves an iterative algorithm for a nonlinear optimisation.

In the `ident` GUI Butterworth low-pass, high-pass and band-pass filters of a specified order can be applied to input and output data. In many cases that should provide enough means to tune a model fit. If for some reason another prefilter $L(z)$ is needed, then this can always be applied at MATLAB's command prompt by computing

$$u_F(t) = L(z)\,u(t) \quad \text{and} \quad y_F(t) = L(z)\,y(t), \tag{6.66}$$

and next using the filtered data for the identification. The `idfilt` command can be useful for this purpose.

Conclusion

The main conclusion from this section is that there are several ways in which the outcome of the identification can be influenced. Of course the estimated model depends on the chosen model set. In addition Eq. (6.65) shows that the result can be tuned by

- the chosen input spectrum $\Phi_u(\omega)$,
- the chosen prefilter $L(z)$ and
- (when possible) the chosen noise model $H(z)$, which is actually complementary to $L(z)$.

Approximate modelling: time domain analysis

In addition to the insight obtained from a frequency domain analysis of the outcome of a PEM model estimation, there are also some statements that apply directly to the time domain data.

In the discussion of the step responses of Fig. 6.3(d) it was already noticed that for the considered example the ARX model did not yield a correct overall step response, but that the initial response during the first few samples was correct. This is reflected in the following properties that are stated without proof.

- The asymptotic estimator θ^* of an ARX model (order denominator = n_a, order nominator = n_b-1, delay $n_k = 0$) with an input signal that satisfies $R_u(\tau) = 0$ for $\tau \neq 0$ is *always* a stable $G(z, \theta^*)$ that gives an exact result for the first n_b samples of the impulse response $g_0(k)$, $k = 0, ..., n_b - 1$.
- On the contrary, in an OE model the (total) squared difference between the impulse responses of system and model are minimised.

To illustrate these statements, the previously introduced fourth order system is used once more. Again models of too low order are estimated. Figure 6.6 shows the impulse responses of the exact G_0, three ARX systems (`arx120`, `arx230` and `arx340`) and two OE systems (`oe210`, `oe320`). The left graph focusses on the first five time samples and the right graph shows the full nonzero part of the impulse response of G_0.

In agreement with the statement above, the impulse response of the `arx230` model ($n_b = 2$) agrees well with the exact result for $t = 1$ and $t = 2$. At later time instances the identified model is different from the correct result and the overall impulse response differs significantly from the impulse response of G_0.

In the `arx340` model, $n_b = 3$ and not only the first three samples are correct, but even the overall impulse responses of `arx340` and G_0 agree quite well. This result illustrates that there is a guarantee that the first n_b samples of the impulse response are correct, but nothing is known about the rest of the response. It may be bad like for the `arx230` model, it may also be quite good like for the `arx340` model.

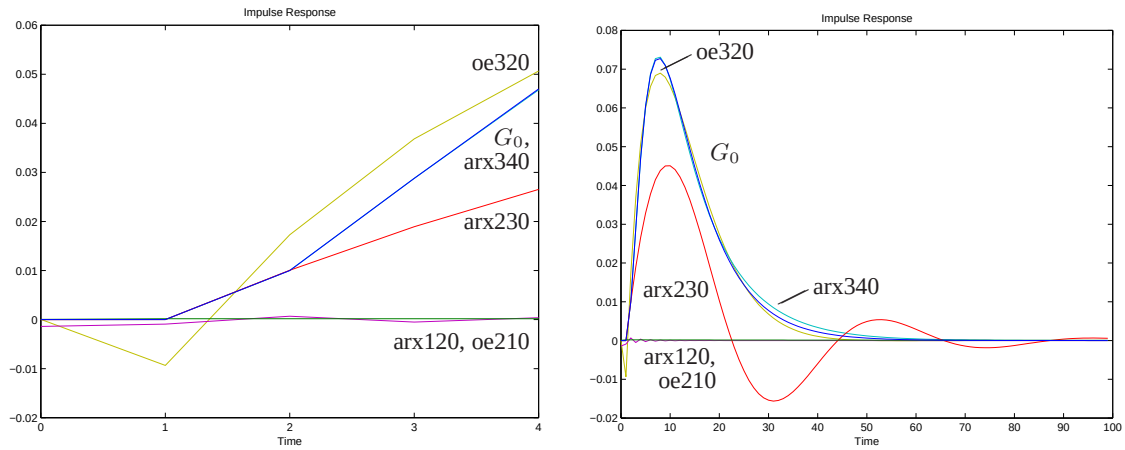


Figure 6.6: Impulse responses: (a) Initial response of the first 5 samples and (b) the overall response during 100 samples.

On the other hand, decreasing n_b seriously deteriorates the fit result. The impulse response of the `arx120` model does not represent the dynamic behaviour of G_0. Nevertheless, the estimated impulse response at $t = 1$ is zero, which is still correct as it should be. Models of lower order are not considered as taking $n_b \leq 2$ does not give meaningful results.

The differences between the OE models and ARX models is particularly clear when models of the same order are compared, like e.g. `oe320` and `arx230`. The impulse response of the ARX model is only correct for the first time samples. In this range the agreement between `oe320` and G_0 is not that good, whereas the overall correctness of the OE model is much better than for the ARX model.

6.6 Selecting a model and validation

So far this chapter dealt with the choice of the model set and some analysis of the identified models. To define the model set, a parameterisation has to be chosen. For PEM models, Fig. 6.1 shows the assumed reality and the system and noise models, $G(z, \theta)$ and $H(z, \theta)$, are written as polynomial fractions. Next we chose a specific model structure for these transfer functions like ARX, ARMAX, OE, FIR or BJ. The choice can be motivated by considerations about linearity in the parameters (only ARX, FIR) or an independent parameterisation of $G(z, \theta)$ and $H(z, \theta)$ (only OE, FIR, BJ). Finally, the model complexity is chosen, by setting the number of parameters in each of the polynomials or in other words by selecting the desired order of the models.

The next step in the identification is the estimation of the parameters and a model is obtained. Clearly, this can be done for many model sets and the important question is:

> What are the criteria to make the above mentioned choices for the model set in order to obtain a *good* model for a *fair* price?

To answer this question, it has to defined what makes a good model. Actually, this depends on the goal for which the model is obtained. It should for example simulate the most important dynamic behaviour or it should be sufficiently accurate to design a (stable) controller. More general requirements are related to consistency and typical requirements are that the model should have a small bias and a a small variance.

Unfortunately, these requirements are conflicting. In general to obtain a model for a real system with a small bias it is necessary to use a large model set. On the other hand, for a small variance a small number of parameters is preferred.

In other words, a large model set offers the advantage that the fit with the data will be better (small residue). However the variance of the parameters can be large. Even worse, it may appear that some of the parameters depends on the specific noise realisation. This effect is called *overfit* and means that some of the parameters do not provide information about the system, as they are only used to generate some features in the output from the specific noise realisation. Such parameters will change significantly when the experiment is repeated having a different noise realisation.

A small model set on the contrary will show a bad fit with the data (large residue), although the variance of the identified parameters will be smaller. Such models will also make a better distinction between stochastic and structured effects in the data. In general the "best" model will be somewhere "in the middle".

In order to look for models at a *fair* price, we should also know what makes up the "price" of the model? This price may depend on the identification. It may take a long time to find the optimal parameters from a nonlinear optimisation criterion. It may also cost a lot of work to investigate many models, so at some moment one has to stop and make a choice.

The price also depends on the intended usage of the model. For controller design it may not be necessary to investigate high frequent behaviour in all detail. For simulations a high order model may be too expensive to evaluate, in particular when simulations need to be carried out in real-time.

Returning to the question raised at the beginning of this section, we need criteria to evaluate to chose the model set in which our "best" model will be identified. In the remainder of this section we will discuss a number of approaches from which such criteria can result:

- A priori considerations, so before experiments are carried out.
- Analysis of the data from an experiment prior to the actual (parametric) identification.
- A posteriori comparison of the identified models. The models may be identified in different model sets and/or by applying different identification criteria (e.g. with and without prefiltering).
- Validation of the models should give a qualification of the capability of a model to represent the measured data.

These approaches are not a list of considerations that need to be addressed for every identification task. In each case some approaches may be more applicable and in the end all useful insight will be combined. to select the "best" model set and to identify the "best" model.

A priori considerations

Before any experiments are carried out, physical insight of the system can be used to obtain information regarding the (minimal) order of the model and/or regarding the nature of the noise disturbance. This insight provides also information about the expected number of parameters n. From the expressions of the variances of the model estimates in Eqs. (6.47) and (6.48) we know that the number of data points N should be sufficiently large with respect to the number of number of parameters to be estimated, $N \gg n$. As a rule of thumb, it should be at least $N > 10n$, but this rule is not always applicable as the required number of points depends strongly on the signal to noise ratio.

Analysis of the data

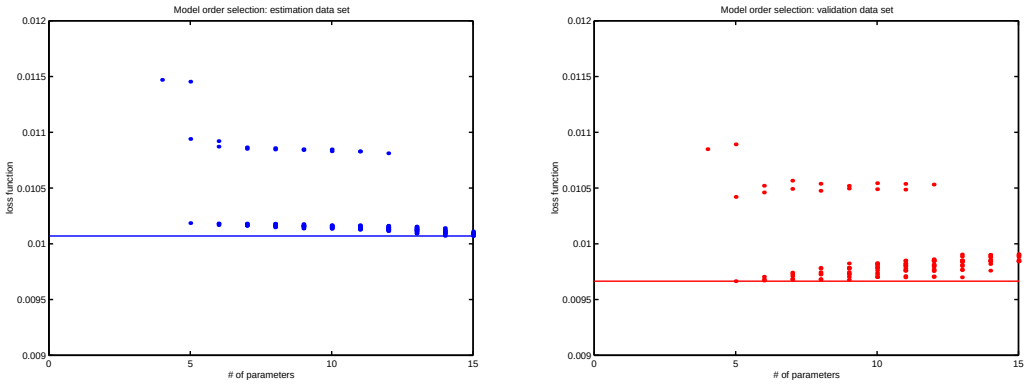
Once a set of measurement data is available, it can be analysed before the actual PEM identification is carried out. Such analysis includes non-parametric identification like spectral analysis. Resonance and anti-resonance peaks may show up that provide information about the model order. Furthermore, changes in the phase can also provide information about the presence of poles and zeros.

As a next step the approximate realisation methods can reveal the order of a system. When a Hankel matrix is available from an impulse response, it can be analysed with SVD. More general, and more practical, SVD can also be applied using input and output data as in a subspace identification. Also in that case the order of the system is estimated without any further investigation of the detailed model structure.

A posteriori comparison of the identified models

Once a number of parametric models have been identified with e.g. subspace and/or PEM techniques, we would like to compare the quality of these models. Before addressing this general case, we first discuss the selection of the "best" model within the same model structure but with varying model complexity, so with a varying number of parameters.

Remembering that out identification criterion aims at the minimisation of the (filtered) prediction error in a least squares sense as in Eq. (6.42). Then an obvious suggestion is to compare the obtained results for the criterion $V_N(\hat{\theta}_N, Z^N)$ as a function of the parameters sets $\hat{\theta}_N$ of different model orders. Figure 6.7(a) shows the outcome of this approach for ARX models. Data has been generated using a system with five parameters ($n_a = 2$, $n_b = 3$). The graph shows the cost function as a function of the number of parameters in the identified model. There are several possible combinations of n_a and n_b to sum up to the same number of parameters, so for each fixed sum $n_a + n_b$ more cost functions are evaluated. In the graph it can be seen that initially the cost function decreases rapidly until a model with five parameters is identified. However, it appears that when more than five parameters are used the cost function continues to decrease. So the best fit is obtained with the most complex model. That is not desirable as the real system can be described with five parameters and the remaining parameters only improve the fit for the specific noise realisation.



(a) With overfit (b) With cross-validation

Figure 6.7: The cost function V_N for ARX models with an increasing number of parameters $n_a + n_b$.

A solution to avoid this *overfit* is the use of *cross-validation*. That means that different data sets are used for the identification and for the validation. In practice it is possible to carry out two experiments, but it is often more convenient to use one data set and to split it into two parts, e.g. two halves. Next the parameters $\hat{\theta}_N$ are identified by minimising the cost function for one part of the data. The quality of the fit with these parameters is evaluated by considering the cost function computed with the identified model in the second part. For the same data set as before this is demonstrated in Fig. 6.7(b). Initially the cost function decreases again quickly until five parameters are used. Then with an increasing number of parameters the cost function increases slowly, indicating that the extra parameters give a better fit for the noise realisation only in the first part, but not in the second part.

It can be concluded from Fig. 6.7(b) that with cross-validation a better guess for the "optimal" number of parameters is obtained. However, the minimum of the cost function near five parameters is

not very distinct and by repeating the procedure with new data sets the best guess is not always found for five parameters.

For ARX models automatic (and more reliable) selection criteria exist [7, pages 3-64 ff, 4-155 ff]. They differ in the exact mathematical expression, but the common approach is that a penalty is added for an increase of the ratio n/N, so an increase of the number of parameters without an increase of the number of data points. These criterion are

- Akaike's Information Criterion (AIC)

$$\text{AIC} = \log((1 + 2\,n/N)\,V). \tag{6.67}$$

- Akaike's Final Prediction Error Criterion (FPE)

$$\text{FPE} = \frac{1 + n/N}{1 - n/N}\,V. \tag{6.68}$$

- Rissanen's minimum description length (MDL)

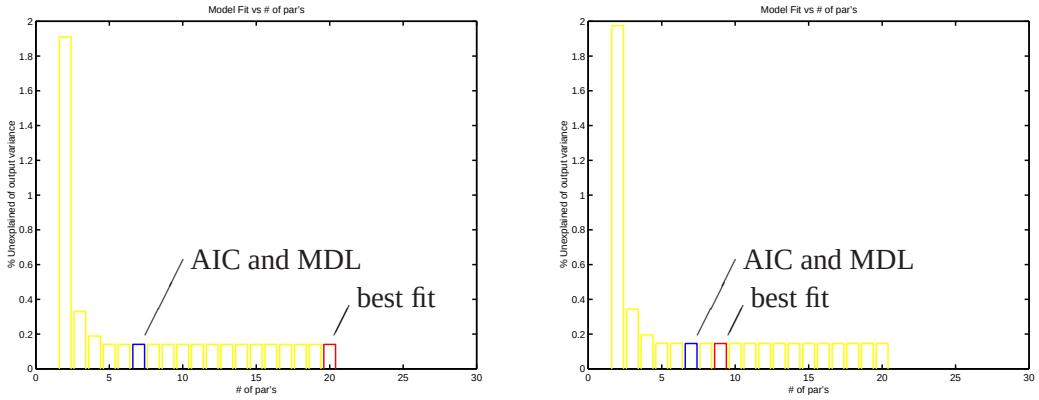
$$\text{MDL} = (1 + \log N * n/N)\,V. \tag{6.69}$$

In the `ident` GUI these criterions are available for a selection of the best model order of an ARX model. It is used when a large number of ARX models are estimated which is possible as the ARX identification is a quick and reliable linear least squares problem. Next the best fit criterion, i.e. smallest V_N, and the above mentioned criterions are applied to suggest a best fit.

Example 1: Fourth order ARX model

The data from the fourth order ARX model with a disturbance of appendix A.2 are used to demonstrate the model estimation for ARX models. This system fits exactly in the `arx442` model set, so it has eight parameters.

Figure 6.8(a) shows the result when identification and validation are carried out on one data set. The best fit is again the model with the largest number of parameters. The AIC and MDL criterions suggest seven parameters. In Fig. 6.8(b) cross-validation is used. For the best fit now only nine parameters are needed and the AIC and MDL criterions suggest again seven parameters leading to an `arx432` model.



(a) Overfit (b) With cross-validation

Figure 6.8: The ARX order estimation for data of an fourth order ARX model with eight parameters. The best fit according to the MDL and AIC criterion and according to a best fit are indicated.

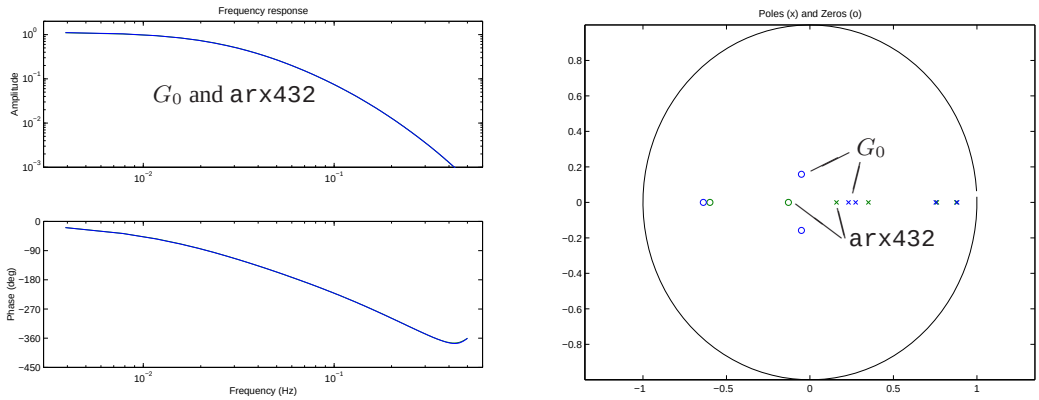
At first sight, the suggestion for the best model being an `arx432` model may be somewhat surprising as the real system is an `arx442`. The real system is

$$G_0(z) = \frac{0.001z^{-2}(10 + 7.4z^{-1} + 0.924z^{-2} + 0.1764z^{-3})}{1 - 2.14z^{-1} + 1.553z^{-2} - 0.4387z^{-3} + 0.042z^{-4}} \tag{6.70}$$

and the identified `arx432` model equals

$$G_{\text{arx432}}(z) = \frac{0.001z^{-2}(10 + 7.3z^{-1} + 0.8z^{-2})}{1 - 2.147z^{-1} + 1.557z^{-2} - 0.4308z^{-3} + 0.0371z^{-4}}. \tag{6.71}$$

The last term in the nominator of G_0 can not be identified in the `arx432` model set, so it is discarded in the identified model. Comparing the coefficients in the nominator of G_0 it can be seen that this term has the smallest coefficient, so the term with the smallest contribution to the nominator is not taken into account. In Fig. 6.9 the spectral analysis and the pole-zero plot of the real system G_0 and the identified `arx432` are compared. In the pole zero plot there are of course differences as the `arx432` has one zero less than G_0. However, in the spectral analysis there are no noticeable differences, so the two models are very similar. Apparently it is quite difficult to estimate all eight parameters from the available data and more samples should be recorded to obtain a good estimate for the eighth parameter.



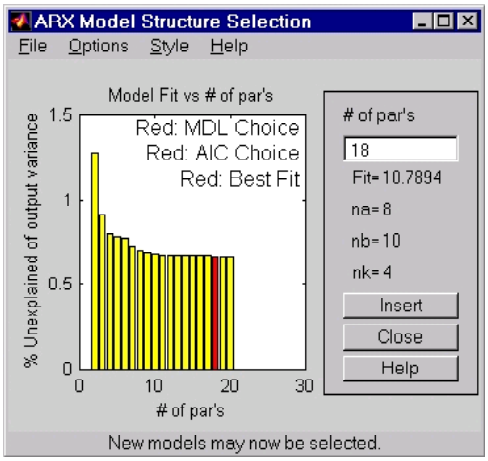
(a) Spectral analysis (b) Pole-zero plot

Figure 6.9: Analysis of the "best" `arx432` identified model and the real system G_0.

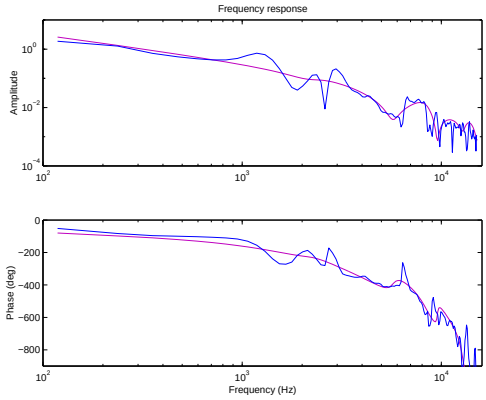
Example 2: Piezo data

On real data the selection of the best model may be more complicated. Figure 6.10 shows the analysis of the data from the piezo mechanism, appendix A.3. For such data the order selection may suggest quite large model sets even if cross-validation is applied. According to the ARX order selection Fig. 6.10(a) an `arx 8 10 4` model should give the best result for this data. This model is compared with a non-parametric spectral model, `etfe60`, in Fig. 6.10(b). Clearly, the ARX model has the typical strong emphasis on the high frequencies. The two common approaches to obtain a better low frequency fit are increasing the model order and/or filtering. A larger model order is not desirable as a low order model is preferred. Filtering appears to be only partially successful (not shown in the graph). Then there is one addition to filtering: resampling of the data. That means that data points are removed, e.g. only every fourth sample is kept and the rest is discarded. Effectively that means that the sample frequency and thus the Nyquist frequency are decreased. Consequently, the identified models can not contain the original high frequencies anymore. Note that filtering has to be applied *before* resampling to avoid aliasing effects. The MATLAB command `resample` can deal with this.

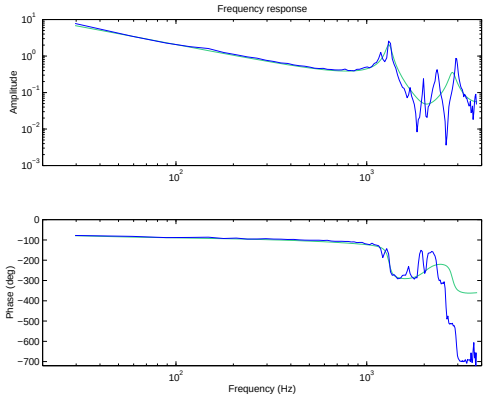
The resample rate depends on the original Nyquist frequency and on the desired frequency range of the identified models. The piezo data has been resample with a factor 4, thus decreasing the Nyquist frequency from 15.2 kHz to 3.8 kHz. Before resampling low-pass filtering is applied to reduce the power



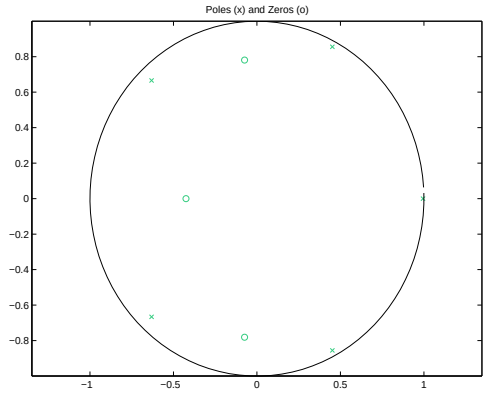
(a) ARX order selection.



(b) Spectral analysis of `arx 8 10 4` (smooth) and `etfe 60`.



(c) Spectral analysis after resampling of `arx542` (smooth) and `etfe 200`.



(d) Pole-zero plot of `arx542`.

Figure 6.10: Analysis of ARX models of the piezo data.

above this frequency. For the resampled data the ARX order selection suggests lower order model. Figure 6.10(c) shows the spectral analysis of an `arx542` model. It is compared to a non-parametric spectral model `etfe200` that is also estimated from the resampled data. Both spectrums show much more detail at lower frequencies. Clearly, the `arx542` model can describe the first eigenfrequency quite well. The `etfe200` model shows a lot more resonances and anti-resonances at higher frequencies. If those should be modelled by the ARX model, again a more complex model is needed of course. If only the first eigenfrequency should be identified correctly, the `arx542` does a good job. In the pole-zero plot of Fig. 6.10(d) the poles of the first eigenfrequency are clearly visible. Analysing the model in more detail reveals that the eigenfrequency is ≈ 1300 Hz with a relative damping of $\zeta \approx 0.03$.

Validation of the models

In the discussion of the models so far we have compared the identified models with each other e.g. by evaluating the cost function of each model. What remains unclear is a definite answer to the question whether an identified model is really "good enough". And if a model is good enough then the chosen model set is also appropriate for the system at hand.

The available techniques for this purpose are

- Comparison of model $G(z, \hat{\theta}_N)$ with previous results. For the simulated data, the identified models were compared with the known data generating system. For measured data that is of course

not possible. In the analysis of the piezo data, the use of a non-parametric spectral analysis proved to be very useful to qualify the parametric models. Missing dynamics could be traced, but on the other hand the qualification of the model remains subjective as it depends on the user's preference to decide what dynamics should be present in the model.

- Model reduction in the case of pole-zero-cancellation. If the pole-zero plot reveals that one or more poles cancel with zeros (within the relevant confidence interval), this may indicate too high model order. Reducing the order of the polynomials may give a model that performs equally well, but this has always to be verified.

- It has not yet been outlined, but for all identified parameters confidence intervals can be computed. Large confidence intervals for one or more parameters may indicate too high model order (or too few data).

- Simulation of the model: Consider the simulation error

$$e_{\rm sim}(t) = y(t) - y_{\rm sim}(t) = y(t) - G(z, \hat{\theta}_N)\, u(t). \tag{6.72}$$

  For the optimal fit, the simulation error equals only the disturbance, so $e_{\rm sim}(t) = v(t) = H_0(z)e(t)$. When the model is not optimal, the simulation error will also include model errors, so it will be larger. For that reason the quality of identified models can be investigated by looking at the average size of the simulation error. The difficulty is to specify the allowable size of this error as that equals the (unknown) size of the disturbance v.

  Furthermore, in order to avoid overfit, cross-validation is preferable.

- Residual test: Check that the prediction error is indeed a white noise signal!

The latter technique will prove to be very useful. Recall that the model parameters are found (mostly) by minimising the least squares sum of the prediction error. According to the assumed model structure this residual error should converge to a white noise signal. Two tests are available for this check:

1. Is the prediction error $\varepsilon(t, \hat{\theta}_N)$ a realisation of white noise? This can be verified by evaluating the autocovariance $R_\varepsilon^N(\tau)$. For a white noise signal the autocovariance should be small for $\tau \neq 0$ and sufficiently large N, so $R_\varepsilon^N(\tau) \to \delta(\tau)$. For a finite number of data samples, the autocovariance is estimated as

$$R_\varepsilon^N(\tau) = \frac{1}{N} \sum_{t=1}^{N-\tau} \varepsilon(t+\tau)\varepsilon(t) \quad (\text{for } \tau \geq 0). \tag{6.73}$$

   The requirement that $R_\varepsilon^N(\tau)$ should be small for $\tau \neq 0$ means that

$$\frac{|R_\varepsilon^N(\tau)|}{R_\varepsilon^N(0)} \leq \frac{N_\alpha}{\sqrt{N}}, \tag{6.74}$$

   where N_α is the confidence level for the confidence interval with probability α, e.g. $N_{95\%} = 1.96$, $N_{99\%} \approx 3$. Equation (6.74) is a test for both the system model $\hat{G}$ and for the noise model $\hat{H}$.

2. Is the prediction error $\varepsilon(t, \hat{\theta}_N)$ a realisation of a stochastic process? This can be verified by evaluating the cross-covariance $R_{\varepsilon u}^N(\tau)$. The prediction error should not be correlated to the input signal, so the cross-covariance should be small for any τ: $R_{\varepsilon u}^N(\tau) \to 0$. For a finite number of data samples, the cross-covariance is estimated as

$$R_{\varepsilon u}^N(\tau) = \frac{1}{N} \sum_{t=1}^{N-\tau} \varepsilon(t+\tau)u(t) \quad (\text{for } \tau \geq 0). \tag{6.75}$$

Its value is sufficiently small, if

$$R_{\varepsilon u}^N(\tau) \leq \frac{N_\alpha \sqrt{P}}{\sqrt{N}}, \tag{6.76}$$

where N_α is the confidence level as before and P can be estimated from

$$\hat{P}^N = \sum_{k=-\infty}^{\infty} \hat{R}_\varepsilon^N(k) \hat{R}_u^N(k). \tag{6.77}$$

Equation (6.76) is a test for only the system model $\hat{G}$.

The big advantage of these residual tests is that they test a model by only using the experimental data. It is tested directly whether the identified model can represent the data within specified confidence levels or not.

Example: Fourth order ARX model

The residual tests are demonstrated with the data from the fourth order ARX model with noise (`arx442` model). The ARX order estimation of Fig. 6.8 suggested an `arx432` model. With residual tests it can be verified if this is indeed a "correct" model.

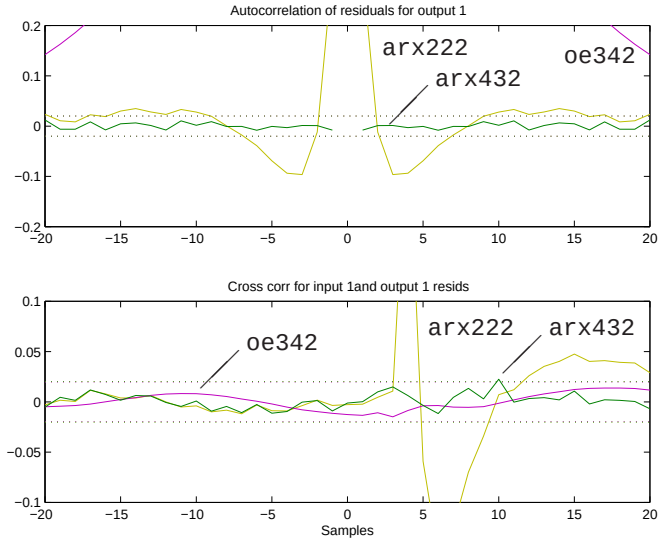


Figure 6.11: Residual tests for models identified from data generated by an `arx442` system.

Figure 6.11 shows the results. For the `arx432` model both the autocovariance and the cross-covariance tests are within the confidence bounds indicating that both the system and noise models are acceptable. In the discussion near Eq. (6.71) it was already made plausible that due to noise in the data one term in the nominator of the original $G_0(z)$ could not be estimated. Now the residual tests confirm that conclusion. If more data would be available (larger N), the confidence levels for the residual tests would be closer to zero and it may appear that the `arx432` model is no longer acceptable.

In Fig. 6.11 also an `oe342` model is tested for which the noise model can never match the data generating system. The autocovariance plot indeed hardly fits on the scale of the top graph. Nevertheless, the cross-covariance plot in the lower graph remains well within the confidence bounds, so the identified system model is also acceptable. A poor fit is obtained for a low order `arx222` model for with neither covariance plot is within the confidence limits.

7 Experiment design

7.1 Introduction

In chapter 6 we have seen that for the identification of a system we need measurement data, a model set, an identification criterion and the tools to obtain the identified model. The outcome of the identification can be tuned by the user by choosing

- the model set (model structure and order),
- a prefilter $L(q)$,
- the experimental conditions like the number of data samples N, the sample time T_s, the input spectrum $\Phi_u(\omega)$, etcetera.

The first two ways to tune the identified model are applied *after* the measurements have taken place and are discussed in chapter 6. The latter aspects have to be considered *beforehand* and will be addressed in this chapter.

Before doing that we recall the requirements that should be satisfied by the identified model. Generally spoken, consistency is an important property. It is difficult to be sure that the parameter estimate is consistent as the requirement that the real system has to fit in the chosen model set may be hard to realise. A second requirement is that the input signal needs to be sufficiently exciting. The design of input signals will be discussed in this chapter.

Of course, the model should also be adequate for the intended purpose, which may be simulation, prediction, diagnostics, controller design or some other goal. For any of these applications it may be necessary to emphasise the accuracy of the model in a specific frequency range. The applied input signal has to assure that these frequencies are indeed excited.

The frequency response or Bode plot describes the bahaviour of a system in the frequency domain. Models are similar when their frequency response is similar. To understand the outcome of the identification in the frequency domain, we recall that it was found that the asymptotic estimator of the parameters θ^* minimises

$$\bar{V}_F(\theta) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \{|G_0(e^{i\omega}) - G(e^{i\omega},\theta)|^2\,\Phi_u(\omega) + \Phi_v(\omega)\}\,\frac{|L(e^{i\omega})|^2}{|H(e^{i\omega},\theta)|^2}\,\mathrm{d}\omega \tag{7.1}$$

in which a compromise between two mechanisms plays a role:

- Minimisation of

  $$\frac{1}{2\pi}\int_{-\infty}^{\infty} |G_0(e^{i\omega}) - G(e^{i\omega},\theta)|^2\,Q(\omega,\theta^*)\,\mathrm{d}\omega,$$

  with

  $$Q(\omega,\theta) = \Phi_u(\omega)\,|L(e^{i\omega})|^2\,/\,|H(e^{i\omega},\theta)|^2.$$

- Fitting of $|H(e^{i\omega},\theta)|^2$ with the error spectrum.

The limit θ^* depends on the model set, the prefilter $L(q)$ and the input spectrum $\Phi_u(\omega)$. These are the design variables for the estimation of the transfer functions.

When two models should be similar in the frequency domain that is often interpreted as a similarity between the frequency responses or Bode plots. If that is considered to be the goal of the fitting of the identified model with the exact $G_0(e^{i\omega})$, some care has to be taken in comparing the fit criterion and the visual appearance of the usual Bode plots.

The vertical axis of the Bode amplitude plot shows $\log|G|$, which is approximated better with a small *relative* error. We are minimising

$$\frac{1}{2\pi}\int_{-\infty}^{\infty} \left|\frac{G_0(e^{i\omega}) - G(e^{i\omega},\theta)}{G_0(e^{i\omega})}\right|^2 |G_0(e^{i\omega})|^2 Q(\omega,\theta^*)\,\mathrm{d}\omega, \tag{7.2}$$

so $Q(\omega,\theta^*)$ should be large when $G_0(e^{i\omega})$ becomes small, i.e. at high frequencies.

The horizontal axis shows $\log\omega$, whereas the data on the frequency axis is available in discrete frequencies that are equally distributed on a linear scale. This means that there are relatively more data at high frequencies and errors at these high frequencies will dominate. This can be compensated by looking at

$$|G_0(e^{i\omega})|^2\, Q(\omega,\theta^*)\,\mathrm{d}\omega = \omega|G_0(e^{i\omega})|^2\, Q(\omega,\theta^*)\,\mathrm{d}(\log\omega) \tag{7.3}$$

So the fit at low frequencies improves if $|G_0(e^{i\omega})|^2\, Q(\omega,\theta^*)$ is larger than ω in that frequency region.

The factor Q can be manipulated by modification of the *design variables*: input spectrum $\Phi_u(\omega)$, noise model set $\{H(e^{i\omega},\theta)\}$, and prefilter $L(q)$.

7.2 Experiment design

For the preparation of an identification experiment, a first analysis of the process is performed and typical questions to be addressed are:

- What are the possible input(s) and output(s) of the system?
- How are these signals being measured?
- At which frequencies do we expect essential dynamic behaviour?
- What are usual and extreme amplitudes of the signals.

Not all answers may be readily available. Preliminary experiments can provide more insight in the system's behaviour. These experiments do not specifically aim at identification, but cover e.g.:

- Measurement of output signal for constant input signal to determine the noise level.
- Measurement of a step response to investigate:
  - Linearity
  - Relevant frequencies
  - Static gain
  - Selection of input signal

  A "staircase" signal is a sequence of steps that can both increase and decrease. It can reveal more details of non-linear behaviour and e.g. hysteresis.
- Non-parametric identification (correlation and frequency analysis) with a harmonic or broadband input signal to determine:
  - Relevant frequencies
  - Duration of the experiment
  - Sample frequency (section 7.4)
  - Selection input signal

7.3 Input signals

Quite often a broadband input signal is desirable. It is sufficiently exciting and the data contains information of the system in a large range of frequencies. In this section some often applied input signals are described. All signals can be created with the MATLAB command `idinput`:

```
u = idinput(N,type,band,levels)
```

The possibilities for `type` are:

`RGS`: Random, Gaussian Signal: discrete white noise with a flat spectrum.

`RBS`: Random, Binary Signal (default).

`PRBS`: Pseudo-Random, Binary Signal.

`SINE`: sum of harmonic signals (sine functions).

7.3.1 Pseudo-Random, Binary Signal

The `PRBS` signal is a signal that can be generated easily by a computer algorithm using n shift registers and *modulo-2 addition* $\oplus$ (see also figure 7.1):

$$\begin{array}{rcll} s(t) & = & x_n(t) & \\ x_i(t+1) & = & x_{i-1}(t) & 2 \leq i \leq n \\ x_1(t+1) & = & a_1 x_1(t) \oplus a_2 x_2(t) \oplus ... \oplus a_n x_n(t) & \end{array} \tag{7.4}$$

for a given initial condition and binary coefficients $a_1, ..., a_n$.

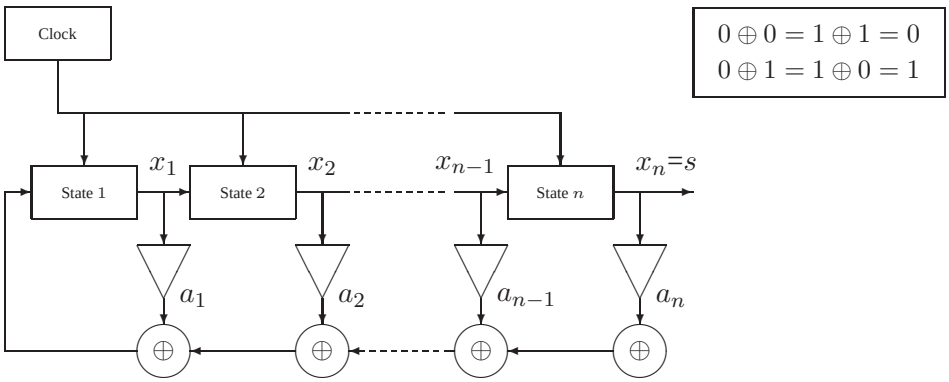


Figure 7.1: The generation of a PRBS signal.

The output signal is a deterministic periodic signal. It is called "pseudo-random" as it exhibits some properties that approximate a random signal. For this reason, it is desirable that the signal is a "Maximum length PRBS", which means that the period of the PRBS is as large as possible. This maximum period equals $M = 2^n - 1$ samples and a PRBS is a maximum length PRBS if the coefficients are chosen correctly. Examples are

n	$a_i = 1$	n	$a_i = 1$	n	$a_i = 1$	n	$a_i = 1$
3	$1, 3$	7	$1, 7$	11	$9, 11$	15	$14, 15$
4	$1, 4$	8	$1, 2, 7, 8$	12	$6, 8, 11, 12$	16	$4, 13, 15, 16$
5	$2, 5$	9	$4, 9$	13	$9, 10, 12, 13$	17	$14, 17$
6	$1, 6$	10	$3, 10$	14	$4, 8, 13, 14$	18	$11, 18$

(7.5)

where the numbers indicate for which i the coefficients a_i are non-zero, i.e. 1, and n is as before the number of shift registers.

The binary signal $s(t)$ generated by equation (7.4) has output values 0 and 1. It can be transformed into a signal $u(t)$ with amplitude c and mean m with

$$u(t) = m + c(-1 + 2\,s(t)). \tag{7.6}$$

It can be shown that the following properties hold for this signal:

$$\begin{aligned} \bar{E}u(t) &= m + \tfrac{c}{M} \\ R_u(0) &= (1 - \tfrac{1}{M^2})c^2 \\ R_u(\tau) &= -\tfrac{c^2}{M}\,(1 + \tfrac{1}{M}) \qquad\qquad \tau = 1, ..., M-1 \end{aligned} \tag{7.7}$$

The autocovariance $R_u(\tau)$ is visualised in figure 7.2. For $M \to \infty$ the autocovariance $R_u(\tau)$ converges to a white noise autocovariance. Similarly, the power spectrum is flat.

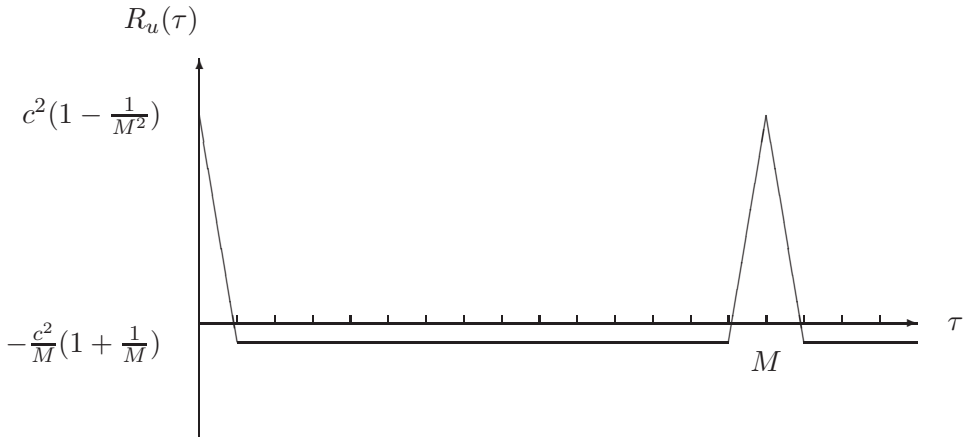


Figure 7.2: Autocovariance of a maximum length PRBS.

The power spectrum can be modified by filtering of the signal with a (linear) filter, but then the binary character is no longer guaranteed. That is a drawback as the binary character is advantageous in the case of (presumed) non-linearities as it has a maximum power for a limited amplitude. It is possible to maintain the binary character and to modify the spectrum by "stretching" the signal as illustrated in figure 7.3. It means that the actual clock period is taken equal to a multiple of the sample frequency, so a PRBS $u(t)$ is transformed into

$$u_{N_c}(t) = u(\mathrm{ent}(t/N_c)), \tag{7.8}$$

where $\mathrm{ent}()$ is the entier (or floor) function that rounds to the nearest integers less than or equal to its argument. Figure 7.4 shows the power spectrum of this signal for some values of N_c. Obviously for $N_c \neq 1$ the spectrum is no longer flat. It appears that with increasing N_c the lower frequencies dominate. Note that there are some frequencies for which the signal provides no input, i.e. $\Phi_u(\omega) = 0$.

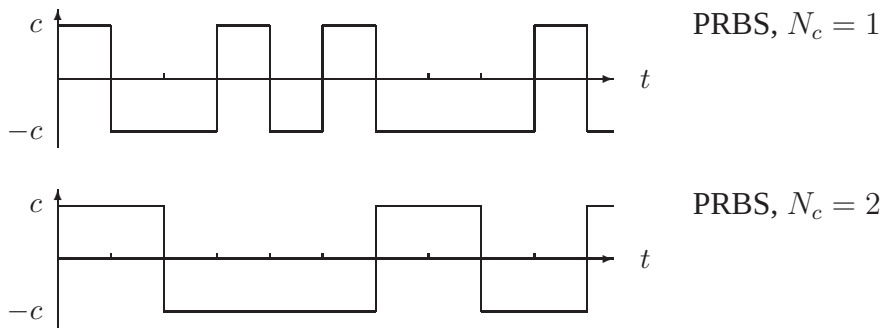


Figure 7.3: Stretching of a PRBS signal.

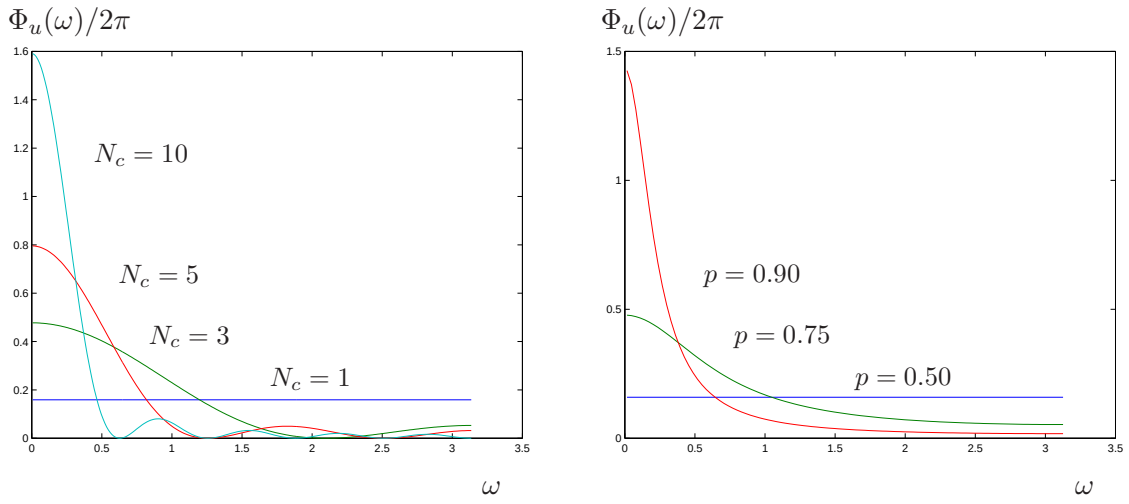


Figure 7.4: On the left the power spectra of a PRBS for some values of the stretching parameter N_c. The right graph shows the power spectrum of some RBS with varying probability p.

7.3.2 Random Binary Signal

The RBS is also a binary signal, but it is a stochastic signal instead of the deterministic definition of the PRBS. Two classes can be distinguished:

1. Generated the signal from

$$u(t) = c \ \text{sign}[R(q)u(t-1) + w(t)], \tag{7.9}$$

in which $w(t)$ is a stochastic white noise process, and $R(q)$ is a stable linear filter. The power spectral density $\Phi_u(\omega)$ depends on $R(q)$. For $R(q) \equiv 0$ the output spectrum is flat.

2. A random binary signal $u(t)$ that equals $\pm c$, according to

$$\begin{aligned} &Pr(u(t) = u(t-1)) = p \quad \text{and} \\ &Pr(u(t) = -u(t-1)) = 1 - p, \end{aligned} \tag{7.10}$$

in which p is the probability that the signal does not change the next sample time ($0 < p < 1$). The power spectral density depends on p. For $p = 1/2$ the spectrum is flat. Figure 7.4 shows some power spectra for other values of p. If $p > 0.5$ the lower frequencies dominate.

7.3.3 Periodic sum of harmonic signals (sine functions)

A "multi-sine" is computed as

$$u(t) = \sum_{k=1}^{r} \alpha_k \sin(\omega_k \, t + \varphi_k), \tag{7.11}$$

with a user defined set of excitation frequencies $\{\omega_k\}_{k=1,\ldots,r}$ and associated amplitudes $\{\alpha\}_{k=1,\ldots,r}$. The phases $\{\varphi\}_{k=1,\ldots,r}$ are usually chosen to obtain a minimal amplitude in the time domain, i.e. *Schroeder-phased sines*. Instead of a complicated search for such optimum set of phases, they can be randomly chosen. The most optimal multi-sine is selected from a number of such tries.

Obviously, when a system is excited with such a multi-sine, information is only obtained in a limited number of frequencies.

7.4 Sample frequency

For the selection of the sample frequency $\omega_s = 2\pi/T_s$ we consider two different situations, namely data acquisition on the one hand and identification or application of the model.

The general advice for data acquisition is to collect as many data as possible. In practice, there will be limitations. For a fixed total measurement time $T_N = N \cdot T_s$ and unlimited number of samples N, the sample time T_s is as small as possible. For a fixed (or limited) number of samples N, the sample time T_s should not be too small in order to capture also the slowest responses of the system. Some rules of thumb are:

- Upper limit T_s, lower limit ω_s:
  The Nyquist frequency $\omega_N = \omega_s/2$ should be well above highest relevant frequency. E.g. for a first order system with bandwidth ω_b: $\omega_s \geq 10\omega_b$.

- As a lower limit for the measurement time T_N it should be taken 5–10 times the largest relevant time constant.

For any sample time T_s, it must be considered to apply (analog) anti-aliasing filters when needed.

For parametric identification or application of the model, one has to realise that numerical aspects define an upper limit for ω_s. A continuous system with state space matrix A_c and ZOH discretisation has a discrete state space matrix $A_d = e^{A_c T_s}$. For $T_s \to 0$ it appears that $A_d \to I$, so all poles of A_d cluster near $z = 1$. In other words, the physical length of the range of the difference equation becomes smaller.

Note further that PE-methods also emphasise high frequencies if T_s is too small. The prediction horizon (one step ahead) becomes small. Furthermore, for smaller T_s so larger ω_s, more higher frequencies are included in the minimisation of

$$\int |G_0(e^{i\omega}) - G(e^{i\omega}, \theta)|^2 \, Q(\omega, \theta) \, \mathrm{d}\omega.$$

In e.g. an ARX model it is very likely that the weighting of (too) high frequencies is too large as

$$Q(\omega, \theta) = \Phi_u(\omega) \, |A(e^{i\omega}, \theta|^2.$$

As a rule of thumb for the upper limit of the sample frequency ω_s for a first order system with bandwidth ω_b is: $\omega_s \leq 30\omega_b$. Combined with the lower limit the sample frequency should be in the range $10\omega_b \leq \omega_s \leq 30\omega_b$ (for a first order system): .

In summary the proposed strategy is to use a high sample frequency ω_s for data acquisition and preliminary experiments. For parametric identification a reduction of ω_s may be needed and this can be accomplished easily by applying a digital filter.

7.5 Data processing

After the data is collected, some further processing may improve the quality of the identification.

- Outliers or spikes can seriously deteriorate the outcome of a least squares fit. They can be found by visual inspection of the data and should be removed either manually or by some automated procedure.

- Means and drift are not included in the (linear) models. They have to be removed of should be modelled explicitly.

- Scaling of signals is not necessary for SISO systems, but is very important for MIMO systems.

- Delays may arise from several sources including the measurement equipment or the ZOH discretisation. A known delay can be removed from the data by compensation. In time discrete models the delay can easily be modelled explicitly, e.g. the parameter n_k in PE models. In either case it has to be assured that the delay is not overestimated as this would result in a-causal models.

- Filtering is important before and after data acquisition. If necessary analog low-pass filters should be used as an anti-aliasing and/or noise filter during the measurement. The cut-off frequency of an (analog) anti-aliasing filters should match the original sample frequency.

  The sample frequency for the identification can be adapted by resampling after applying a digital anti-aliasing filter (explicit or automatic in a MATLAB command). If it is known beforehand that such resampling will take place, then this should be considered when the input data is generated (e.g. by setting the N_c for a PRBS). It is useless to put energy in a frequency band that is later removed.

10 System indentification in the frequency domain

10.1 Introduction

So far we mainly focussed on system identification techniques using time domain data. In chapter 6 the Prediction Error identification Methods (PEM) were discussed. It was shown that although we are using *time domain* data, the obtained models can be qualified quite well in the *frequency domain* with Eq. (6.56). The estimated models can be tuned in the frequency domain by filtering input and output signals with a prefilter $L(z)$ as shown in Eq. (6.65). From the discussion in that chapter it is evident that the frequency response of the system and its models provide valuable insight.

The analysis in the frequency domain can also be applied directly using frequency domain data. This frequency domain data can be the spectral estimate from a non-parametric identification (section 3.3) applied to any time domain data. However, preferably dedicated experiments are carried out to obtain the frequency response. Typically the system is excited with a signal composed from a number of harmonic signals, i.e. a multi-sine excitation e.g. from `idinput`. For the measurement dedicated hardware can be applied like a spectrum analyser. Alternatively, time series of the input and output are recorded and subsequently Fourier transformed. In either case a frequency response of the system is estimated in the frequencies that are present in the input signal.

Compared to the identification in the time domain, this approach offers some inherent advantages. As the input and output signals are only analysed at specific frequencies, the amount of data is reduced significantly from the number of time domain samples to the number of considered frequencies. Noise reduction is applied at this stage also, as the noise in all other frequencies is ignored. The filtering that is applied in the time domain to tune the model estimate in the frequency domain can be applied straightforwardly to frequency domain data by multiplying the data with a weighting function. The use of a well-defined set of frequencies in the input signal allows the detection of non-linear behaviour as this can result in frequencies in the output signal that are not present in the input signal. Finally, frequency domain identification can deal equivalently with time continuous as with time discrete models.

10.2 Frequency domain identification

For frequency domain identification we assume that the data is available as the system's frequency response $\breve{G}(\omega_k)$ at a number of angular frequencies ω_k. Then we have N complex numbers

$$\breve{G}(\omega_k), \quad \omega_k \in \Omega \tag{10.1}$$

representing the amplification and phase shift of the signal at the N angular frequencies

$$\Omega = \{\omega_1, \omega_2, ..., \omega_N\}. \tag{10.2}$$

The "multi-sine" signal mentioned in chapter 7 offers an excitation signal where the frequency grid can be chosen and this is one aspect of the experiment design. The system is excited at these frequencies and the excitation amplitude at each frequency can also be set to obtain optimal signal to noise performance. The frequency response can be computed from the Fourier transforms of input and output signals.

Assuming the data is generated by a LTI system with a time continuous transfer function $G_0(s)$, the measured response can be written as

$$\breve{G}(\omega_k) = G_0(i\omega_k) + V(\omega_k), \tag{10.3}$$

where the frequency response is found by substituting $i\omega$ for s in the transfer function of the real system and there is a disturbance from a stochastic noise process $V(\omega_k)$.

For the identification we define a model set of systems with transfer functions $G(s,\theta)$ that depend on a parameter vector θ. For a LTI SISO the system is parameterised with a transfer function

$$G(s,\theta) = \frac{B(s,\theta)}{A(s,\theta)}, \tag{10.4}$$

where A and B are polynomials in s. Then the identification problem implies that we want to find $\hat{\theta}$ such that it minimises

$$\sum_{k=1}^{N} |\breve{G}(\omega_k) - G(i\omega_k,\theta)|^2 |W(\omega_k)|^2 \tag{10.5}$$

with a chosen weighting function $W(\omega_k)$. Substitution the parameterised transfer function yields the solution expressed as

$$\hat{\theta} = \arg \min_{\theta} \sum_{k=1}^{N} |\breve{G}(\omega_k) - \frac{B(i\omega_k,\theta)}{A(i\omega_k,\theta)}|^2 |W(\omega_k)|^2. \tag{10.6}$$

This optimisation problem resembles the time domain OE-problem, which is non-linear in the parameters θ. It can be solved iteratively, starting with an reasonable initial estimate obtained with a linear regression techniques. Dedicated non-linear optimisation techniques exist like the Sanathanan-Koerner algorithm. Alternatively, it can be solved iteratively by transforming it into an ARX-like problem which is linear in the parameters. This can be accomplished by multiplying the cost function in Eq. (10.6) with the A polynomial. However, this adds weighting with $|A(i\omega_k,\theta)|$ which in turn can be corrected by adjusting the weight as

$$\hat{\theta} = \arg \min_{\theta} \sum_{k=1}^{N} |A(i\omega_k,\theta)\breve{G}(\omega_k) - B(i\omega_k,\theta)|^2 \frac{|W(\omega_k)|^2}{|A(i\omega_k,\theta)|^2}. \tag{10.7}$$

This expression is still non-linear in the parameters θ, unless an estimate A^* is substituted for the A polynomial in the weight

$$\hat{\theta}^* = \arg \min_{\theta} \sum_{k=1}^{N} |A(i\omega_k,\theta)\breve{G}(\omega_k) - B(i\omega_k,\theta)|^2 \frac{|W(\omega_k)|^2}{|A^*(\omega_k)|^2}. \tag{10.8}$$

This optimisation problem is linear in the parameters θ. Now the non-linear optimisation problem (10.6) can be approximated with the following iterative linear procedure:

1. Take some initial estimate for A^*, e.g. $A^*(\omega_k) = 1$ and estimate the parameter set $\theta^{(0)}$ in the polynomials $B^{(0)}(s,\theta^{(0)})$ and $A^{(0)}(s,\theta^{(0)})$ from the linear optimisation problem (10.8).

2. Use the estimated denominator to update $A^*(\omega_k) = A^{(l-1)}(i\omega_k,\theta^{(l-1)})$ and re-estimate $B^{(l)}(s,\theta^{(l)})$ and $A^{(l)}(s,\theta^{(l)})$ from the linear optimisation problem (10.8).

3. Repeat step 2 until some level of convergence is obtained.

There is no guarantee that the parameter set $\theta^{(l)}$ in this iterative procedure converges to the global minimum $\hat{\theta}$ of equation (10.6). Furthermore the following aspects should be taken into account.

The non-linear and linear optimisation problems (10.6) and (10.8) include complex numbers representing the amplification and phase shift of the signal at all considered frequencies. Consequently, the solution $\hat{\theta}$ can include complex numbers as well. This can be avoided by adding negative frequencies. It can be shown easily that each frequency response $\breve{G}(\omega_k)$ for a positive frequency ω_k implies a complex conjugate response $\breve{G}(-\omega_k) = \breve{G}^*(\omega_k)$ at the negative frequency $-\omega_k$ for any system G that is described with real valued parameters. If for all (positive) frequencies, the negative frequencies are also included in the list Ω, it can be seen that the optimisation will give real valued parameters.

In equation (10.4) a time continuous transfer function $G(s)$ is proposed to model system G. Alternatively, a time discrete transfer function $G(z)$ can be considered where of course the computation for the frequency response must be modified. Irrespective the choice for either a time continuous or discrete representation, the data acquisition must be taken into account. In particular, if the system's input for the identification is generated by a digital system with a zero-order hold, then the typical delay $-\omega T_s/2$ introduced by this zero-order hold will appear in the analysis of the recorded data. In the case a time continuous transfer function is used for the identification, this delay is not easily included. Identification is nevertheless possible by adjusting the data for this known delay. More precisely, by using

$$\breve{G}(\omega_k)\ e^{i\omega_k T_s/2},$$

with the sample time T_s for the identification instead of the original $\breve{G}(\omega_k)$.

The weighting function $W(\omega_k)$ must be set to take into account filtering in the frequency domain, e.g. to emphasis some part of the frequency range. Typical weighting functions are

- None.
- The inverse of data, which implies that the *relative* error in the frequency response is minimised.
- A specific frequency range. Note that any filtering can be applied, so even selecting one frequency and ignoring a neighbouring data point.
- The coherence spectrum (see also MATLAB's `help cohere`) which is defined as

$$\hat{C}^N_{yu}(\omega) = \sqrt{\frac{|\hat{\Phi}^N_{yu}(\omega)|^2}{\hat{\Phi}^N_y(\omega)\hat{\Phi}^N_u(\omega)}} \tag{10.9}$$

  It equals one for a perfect correlation between input u and output y, while a zero indicates no correlation between input u and output y, i.e. an output that is dominate by noise.

A Example systems

In these notes a number of sample systems will be used to clarify the theory. This appendix gives the description of these systems.

A.1 Second order ARMAX system

In chapter 2 of the `ident` manual [7, page 2-15] a second order system is introduced. This system will be used in these notes as well to facilitate a comparison between the text in the manual and in these notes.

It is a discrete time system of which the output y is given by

$$y_k - 1.5y_{k-1} + 0.7y_{k-2} = u_{k-1} + 0.5u_{k-2} + e_k - e_{k-1} + 0.2e_{k-2}, \tag{A.1}$$

where u is the known input of the system and e is an unknown white noise disturbance with variance 1. With discrete time transfer functions this can be rewritten as

$$y_k = \frac{q^{-1} + 0.5q^{-2}}{1 - 1.5q^{-1} + 0.7q^{-2}}u_k + \frac{1 - q^{-1} + 0.2q^{-2}}{1 - 1.5q^{-1} + 0.7q^{-2}}e_k, \tag{A.2}$$

so

$$y_k = G_0(q)u_k + H_0(q)e_k, \tag{A.3}$$

with

$$G_0(q) = \frac{q^{-1} + 0.5q^{-2}}{1 - 1.5q^{-1} + 0.7q^{-2}} \tag{A.4}$$

and

$$H_0(q) = \frac{1 - q^{-1} + 0.2q^{-2}}{1 - 1.5q^{-1} + 0.7q^{-2}}. \tag{A.5}$$

As pointed out in Sect. 2.2 these expression can also be written in the z-domain, so

$$y_k = G_0(z)u_k + H_0(z)e_k, \tag{A.6}$$

in which $G_0(z)$ and $H_0(z)$ can be written analogously as $G_0(q)$ and $H_0(q)$ after replacing q by z. Quite often such expressions are rewritten in such a way that only non-negative powers of z remain. After multiplication with z^2 it follows that

$$G_0(z) = \frac{z + 0.5}{z^2 - 1.5z + 0.7} \tag{A.7}$$

and

$$H_0(z) = \frac{z^2 - z + 0.2}{z^2 - 1.5z + 0.7}. \tag{A.8}$$

For a description of the system either set of transfer functions can be used.

In the example of the manuals the sample time $T = 1$ s. In simulations usually $N = 4096$ data points are computed and for u a so-called Random Binary Signal (RBS) of which the frequency band is limited from 0 Hz to 0.3 times the sample frequency f_s, so up to 0.3 Hz. As mentioned above, e is a "white" noise (or random) signal with variance 1. Being random signals, both u and e will be different each time a new simulation is carried out. Throughout these notes a single realisation of these random signals is used. This data set can be downloaded from the web site. It has been computed with

```
% Create the model in iddata format:
modelid = idpoly([1.0 -1.5 0.7], [0.0 1.0 0.5], [1.0 -1.0 0.2]);
Ts=1;
set(modelid, 'Ts', Ts);
% Create a tf model as well:
[A,B,C,D,F]=polydata(modelid);
modelsys=tf(B,A,Ts);

N = 4096;
BW = [0.0 0.3];
u = idinput(N,'rbs',BW);
e = randn(N,1);

y = idsim([u e], modelid);

simul=[y u e];
```

A.2 Fourth order system

A fourth order example system is taken from Ljung [6, Example 8.5]. In most simulations this system does not have a disturbance, so the output of this system is

$$y(t) = G_0(z)u(t) \tag{A.9}$$

with a fourth order system model

$$G_0(z) = \frac{0.001z^{-2}(10 + 7.4z^{-1} + 0.924z^{-2} + 0.1764z^{-3})}{1 - 2.14z^{-1} + 1.553z^{-2} - 0.4387z^{-3} + 0.042z^{-4}}. \tag{A.10}$$

The typical input signal $u(t)$ is a PRBS with 1024 samples with sample time 1 s and variance 1, so $\Phi_u(\omega) \approx 1$.

In some simulations a disturbance is included. In those cases an ARX structure is applied, so the disturbance is generated from a white noise signal with variance $1.0 \cdot 10^{-4}$ and noise model

$$H_0(z) = \frac{1}{1 - 2.14z^{-1} + 1.553z^{-2} - 0.4387z^{-3} + 0.042z^{-4}}. \tag{A.11}$$

In the simulations with a disturbance 32768 samples are simulated.

A.3 Piezo mechanism

Experimental data is available from a mechanical system with a piezo actuator. This mechanism has been manufactured for the course "Project F" in the Bachelor program of Mechanical Engineering. Essentially it is a mass that has to be positioned by the piezo actuator[1]. The position is measured by a sensor and is the output y of the system. The piezo is driven by a electrical current source being the

[1] This system has some resemblance with the experimental setup used for one of the exercises, but they are not identical.

input u. The mechanical system can be modelled as a mass that is fixed to the world by a stiffness. The piezo actuator generates a force parallel to the stiffness that is proportional to the accumulated electrical charge in the actuator, so the integrated current.

This combination leads to a mass-spring system with some damping and an extra integrator. Of course the real system is not as simple as this. The input current is not exactly integrated as there are losses due to electrical resistance. The force displacement relation appears to be non-linear. The system has higher eigenfrequencies and the sensor is also not ideal. Identification should give the correct system behaviour. Figure A.1 gives some typical input output data as is available from the web site. In this case a Pseudo-Random Binary Signal (PRBS) has been used as input.

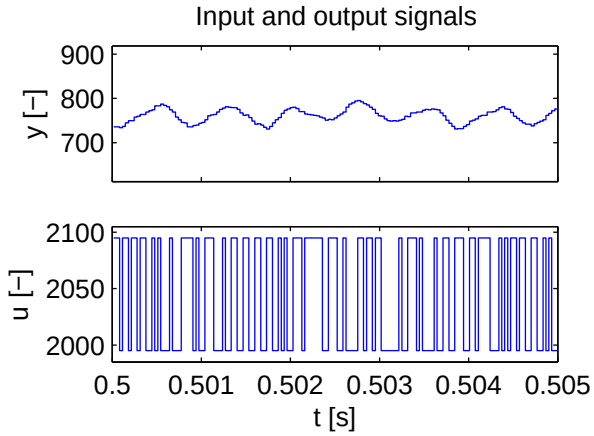


Figure A.1: Example of input and corresponding output signal of a mechanical system.

Bibliography

[1] R.G.K.M. Aarts and J. van Dijk, *Systeem- en Regeltechniek 1* (in Dutch), Lecture notes University of Twente, 2010.

[2] R.A. de Callafon and P.M.J. van den Hof, *FREQID - frequency domain identification toolbox for use with matlab*. Selected Topics in Identification, Modelling and Control (Mechanical Engineering Systems and Control Group, TU Delft), vol. 9, pp. 129-134, 1996.

[3] Gene F. Franklin, J. David Powell and Abbas Emami-Naeini, *Feedback control of dynamic systems*, 6^{th} edition, Prentice Hall, ISBN 9780135001509 (int. ed.), 2009.

[4] Paul van den Hof, *System Identification*, Lecture notes Delft University of Technology, 1997.

[5] Lennart Ljung, *System Identification: Theory for the User*, Prentice-Hall, UT library code WK 62-5:514/518 l065, 1987.

[6] Lennart Ljung, *System Identification: Theory for the User*, 2^{nd} edition, Prentice-Hall, 1999.

[7] Lennart Ljung, *System Identification Toolbox 7 Users Guide*, Revised for Version 7.4.3 (Release 2011b), The MathWorks Inc., September 2011.
WWW: `http://www.mathworks.com/help/pdf_doc/ident/ident.pdf`

[8] The Math Works Inc., MATLAB *7 Getting Started Guide*, Revised for MATLAB 7.13 (Release 2011b), September 2011.
WWW: `http://www.mathworks.com/help/pdf_doc/matlab/getstart.pdf`

[9] The Math Works Inc., *Optimization Toolbox 5 Users Guide*, Revised for Version 6.1 (Release 2011b), September 2011.
WWW: `http://www.mathworks.com/help/pdf_doc/optim/optim_tb.pdf`

[10] The Math Works Inc., *Signal Processing Toolbox 6 Users Guide*, Revised for Version 6.16 (Release 2011b), September 2011.
WWW: `http://www.mathworks.com/help/pdf_doc/signal/signal_tb.pdf`

[11] B.L. Ho and R.E. Kalman, *Efficient construction of linear state variable models from input/output functions*. Regelungstechnik, vol. 14, pp. 545-548, 1966.

[12] H.P. Zieger and J. McEwen, *Approximate linear realizations of given dimansion via Ho's algorithm*. IEEE Trans. Automat. Control, vol. AC-19, pp. 153, 1974.

[13] M. Verhaegen, *Identificaton of the deterministic part of MIMO state space models given in innovations form from input-output data*. Automatica, vol. 30, pp. 61-74, 1994.

[14] Peter van Overschee and Bart de Moor, *Subspace identification for linear systems: Theory, implementation, applications*. Kluwer Academic Publishers, 1996.