Linköping Studies in Science and Technology
Thesis No. 1061

A Survey and Comparison of Time-Delay Estimation Methods in Linear Systems

Svante Björklund

Division of Automatic Control
Department of Electrical Engineering
Linköpings universitet, SE–581 83 Linköping, Sweden
WWW: `http://www.control.isy.liu.se`
Email: `svabj@isy.liu.se`

Linköping 2003

A Survey and Comparison of Time-Delay Estimation Methods in Linear Systems

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

ISBN 91-7373-870-0
ISSN 0280-7971
LiU-TEK-LIC-2003:LIU-TEK-LIC-2003:60

Printed by UniTryck, Linköping, Sweden 2003

To Ulrica

Abstract

In this thesis the problem of time-delay estimation (TDE) in linear dynamic systems is treated. The TDE is studied for signal-to-noise ratios, input signals, and systems that are common in process industry. This also implies that both open-loop and closed-loop cases are of interest. The true time-delay is estimated, which may be different from the time-delay giving the best model approximation of the true system. Time-delays which are not a multiple of the sampling interval are also of interest to estimate.

In this thesis, a review and a classification according to underlying principles of TDE methods in the literature are made. The main classes are: 1) Time-Delay Approximation Methods: The time-delay is estimated from a relation (a model) between the input and output signals expressed in a certain basis. The time-delay is not an explicit parameter in the model. 2) Explicit Time-Delay Parameter Methods: The time-delay is an explicit parameter in the model. 3) Area and Moment Methods: The time-delay is estimated from certain integrals of the impulse and step responses. 4) Higher Order Statistics Methods.

Some new methods and variants of old ones are suggested and evaluated, some of which have good estimation performance and some poor performance. Properties of TDE methods are analyzed, both theoretically and experimentally. Recommendations are given on how to choose estimation method and input signal. Generally, prediction error methods where the time-delay parameter is explicit and is optimized simultaneously with the other model parameters give good estimation quality.

Most evaluations have been conducted with factorial experiments using Monte Carlo simulations in open and closed loop. Some statistical analysis methods have been utilized: The RMS error of the time-delay estimates gives an absolute measure of the performance. ANOVA (ANalysis Of VAriance) and confidence intervals give conclusions with a certain level of confidence.

Acknowledgements

First of all, I would like to thank my supervisor Professor Lennart Ljung for giving my the possibility to join and get some insight into his successful research group and for his skillful guidance in my own research. I am also very grateful to Professor Alf Isaksson, Dr. Alexander Horch and Professor Alexander Medvedev for discussing time-delay estimation and sharing MATLAB code with me. Professor Eva Enqvist, who first introduced me into ANOVA and some other statistical techniques, has been very helpful and always found time for discussions. Thank you. I would like to thank Martin Enqvist, Markus Gerdin, Jonas Gillberg, Frida Gunnarsson, David Lindgren, Dr. Jacob Roll, Ragnar Wallin for reading parts of different versions of the manuscript and giving valuable comments. Thanks to Gustaf Hendeby and others for help with LaTeX and to Jens Larsson and others for help with our computer systems. Ulla Salaneck deserves gratitude for always being helpful with administrative matters. Thanks to all members of the Control and Communication group for creating a nice working atmosphere. This work has been supported by the Swedish Research Council, which is hereby gratefully acknowledged.

Svante Björklund
Linköping, November 2003

Contents

List of Figures

List of Tables

List of Algorithms

Notations

Operators and conventions

$\hat{a}$	An estimation of the quantity a.
$\langle a, b\rangle$	Scalar product of a and b.
$\bar{G}(s)$	Continuous-time system.
$G(z)$	Discrete-time system.
$A^\dagger$	Pseudoinvers of the matrix A.
$\underline{a}$	The vector a without the last element.
$\overline{a}$	The vector a without the first element.
$x \propto y$	x is proportional to y.
(a, b)	The region of the real axis between a and b, excluding the boundaries.
$[a, b]$	The region of the real axis between a and b, including the boundaries.
$\operatorname{Re} a$	The real part of a.
$\operatorname{Im} a$	The imaginary part of a.

Symbols and functions

d	Normalized integer part of the total time-delay T_d.
$\delta(t)$	An impulse function.
ϵ	Subsample part [in number of sampling intervals T_s] of the total time-delay T_d. See T_d.
$\epsilon(t)$	Error in the ANOVA model (Equation 7.17).
$\varepsilon(t)$	Residual in ANOVA, Equation 7.21.
$\varepsilon(t)$,	Prediction error in PEM (Section 4.1.5).
$\varepsilon(t\|\theta)$, $\varepsilon(t\|L)$, $\varepsilon(t\|d,\epsilon)$	Prediction error in PEM for model parameters θ, L and (d,ϵ), respectively.
$\bar{G}(s)$	Continuous-time system including the time-delay. $\bar{G}(s) = \bar{G}_r(s) \cdot e^{-sT_d} = \bar{G}_r(s) \cdot \bar{G}_{\text{ap}}(s)$.
$\bar{G}_r(s)$	Continuous-time system excluding the time-delay.
$\bar{G}_{\text{ap}}(s)$	The continuous-time time-delay seen as an allpass filter.
$G(z)$	Discrete-time system including the time-delay. $G(z) = G_r(z)G_{ap}(z)$.
$\bar{G}_r(s)$	Discrete-time system excluding the time-delay.
$g(t)$	Impulse response of system with time delay $g(t) = g_r(t) * \delta(t - T_d)$.
$g_d(t)$	An approximation of the pure time-delay part $\delta(t - T_d)$ of $g(t)$ (Section 4.1.1).
$g_r(t)$	A system without time-delay.
$g_{\text{ts}}(t)$	Test statistics in CUSUM detection (Section 4.1.3).
$H(s)$	Noise model. Used in closed loop simulations (Section 7.4).
$h(t)$	Absolute threshold in CUSUM detection (Sections 4.1.3, 4.1.6 and Algorithm 2).
h_{std}	Relative threshold in direct and CUSUM detection (Section 4.1.6 and Algorithm 2).
k	Time index in discrete-time. See also t.
$\hat{k}$	Change time estimate in CUSUM detection (Section 4.1.3).
L	Subsample (fractional) part [seconds] of the total time-delay T_d.
N	Number of data in for example PEM.

$N+1$	Number of Laguerre functions in Laguerre domain methods.
$\nu(t)$	Absolute drift in CUSUM detection (Sections 4.1.3, 4.1.6).
ν_{std}	Relative drift in CUSUM detection (Section 4.1.6).
$s(t)$	Distance measure in CUSUM detection (Section 4.1.3).
t	Time index in continuous-time and discrete-time. See also k.
t_a	Detection time in CUSUM detection (Section 4.1.3).
T_d	Total time-delay [seconds]. $T_d = (d+\epsilon)T_s = d \cdot T_s + L$.
T_s	Sampling interval [seconds].
$u(t)$	Input signal.
$V()$	Loss function or function to minimize.
$v(t)$	The noise added to the output of the system (Equation 7.1 and Figure 7.5).
$y(t)$	Output signal. It is also the estimated impulse or step response in direct and CUSUM detection (Section 4.1.6).
$\hat{y}_{\text{std}}(t)$	Estimation of the standard deviation of the estimated impulse or step response (Section 4.1.6 and Algorithm 2).
$\mathcal{Z}^{-1}\{\cdot\}$	Inverse Z-transform.

Abbreviations and acronyms

ANOVA	ANalysis of VAriance (Section 7.5.3)
ARX	Auto Regressive model with eXtra input [Lju99].
CUSUM	CUmulative SUM (Section 4.1.3).
DAP	Discrete-time Allpass Part (Section 4.2.5).
LMS	Least Mean Square [Gus00, GLM01].
OE	Output Error model [Lju99].
PDF	Probability density function.
PEM	Prediction Error Method for system identification [Lju99].
SVD	Singular Value Decomposition [HJ91].
TDE	Time-Delay Estimation.
TIDEA	TIme Delay Estimation Algorithm (Section 5.3.2).
zoh	zero-order-hold sampling [ÅW84].

Time-delay estimation methods

Method	Parameters	Algorithm	Theory
area1	Sec. 7.2	-	Sec. 4.1.1
area2	Sec. 7.2	-	Sec. 6.1
arxdap	Sec. 7.2	-	Sec. 4.2.5
arxdap2	Sec. 7.2	-	Sec. 4.2.5
arxstruc	Sec. 7.2	Alg. 5 (p. 51)	Sec. 5.2.1
arxstruc3	Sec. 7.2	Alg. 5 (p. 51)	Sec. 5.2.1
elnaggar	Sec. 7.2	-	[EDE89] and Chap. 3
firdap	Sec. 7.2	-	Sec. 4.2.5
firdap2	Sec. 7.2	-	Sec. 4.2.5
fischer1	Sec. 7.2	Alg. 3 (p. 41)	Sec. 4.3
fischer2	Sec. 7.2	Alg. 4 (p. 41)	Sec. 4.3
fischer3	Sec. 7.2	Alg. 3 (p. 41)	Sec. 4.3
fischer4	Sec. 7.2	Alg. 4 (p. 41)	Sec. 4.3
fischer5	Sec. 7.2	Alg. 3 (p. 41)	Sec. 4.3
fischer6	Sec. 7.2	Alg. 4 (p. 41)	Sec. 4.3
fischer7	Sec. 7.2	Alg. 3 (p. 41)	Sec. 4.3
fischer8	Sec. 7.2	Alg. 4 (p. 41)	Sec. 4.3
idimp4	Sec. 7.2	Alg. 2 (p. 27)	Sec. 4.1.1-4.1.3, 4.1.6
idimp5	Sec. 7.2	Alg. 2 (p. 27)	Sec. 4.1.1-4.1.3, 4.1.6
idimpCusum3	Sec. 7.2	Alg. 2 (p. 27)	Sec. 4.1.1-4.1.3, 4.1.6
idimpCusum4	Sec. 7.2	Alg. 2 (p. 27)	Sec. 4.1.1-4.1.3, 4.1.6
idproc1	Sec. 7.2	-	Sec. 5.1.1
idproc2	Sec. 7.2	-	Sec. 5.1.1
idproc3	Sec. 7.2	-	Sec. 5.1.1
idproc4	Sec. 7.2	-	Sec. 5.1.1
idproc5	Sec. 7.2	-	Sec. 5.1.1
idproc6	Sec. 7.2	-	Sec. 5.1.1
idproc7	Sec. 7.2	-	Sec. 5.1.1
idstep4	Sec. 7.2	Alg. 2 (p. 27)	Sec. 4.1.1-4.1.3, 4.1.6
idstep5	Sec. 7.2	Alg. 2 (p. 27)	Sec. 4.1.1-4.1.3, 4.1.6
idstepCusum3	Sec. 7.2	Alg. 2 (p. 27)	Sec. 4.1.1-4.1.3, 4.1.6
idstepCusum4	Sec. 7.2	Alg. 2 (p. 27)	Sec. 4.1.1-4.1.3, 4.1.6
lagucont1	Sec. 7.2	-	Sec. 4.2.4
lagudap	Sec. 7.2	-	Sec. 4.2.5
lagudap2	Sec. 7.2	-	Sec. 4.2.5
kurz	Sec. 7.2	-	[KG81] and Sec. 4.1.3, 4.1.6
met1struc	Sec. 7.2	Alg. 6 (p. 51)	Sec. 5.2.3
met1struc3	Sec. 7.2	Alg. 6 (p. 51)	Sec. 5.2.3
moment1	Sec. 7.2	-	Sec. 6.2
moment2	Sec. 7.2	-	Sec. 6.2
oedap	Sec. 7.2	-	Sec. 4.2.5
oedap2	Sec. 7.2	-	Sec. 4.2.5
oestruc	Sec. 7.2	Alg. 9 (p. 54)	Sec. 5.2.2
oestruc3	Sec. 7.2	Alg. 9 (p. 54)	Sec. 5.2.2

1

Introduction

This chapter defines the problem that is treated in this thesis, states the purpose with this work, lists the main contributions in and gives an outline of this thesis.

1.1 The Problem

In this thesis we will study the time-delay estimation (TDE) problem, where we want to estimate T_d in

$$y(t) = G(p)u(t) + n(t) = G_r(p)u(t - T_d) + n(t), \tag{1.1}$$

where the system $G_r(p)$ is a SISO (single-input single-output) time-invariant linear transfer function without time-delay. In signal processing applications, the system is often restricted to be a constant [C^+81, MG^+98], but here $G_r(p)$ will be a transfer function with dynamics, of the type which typical in process industry, see e.g. [IHD01b, Swa99]. This means that both the open-loop and closed-loop cases are of interest and that we will study TDE for SNRs (signal-to-noise ratio), input signals and systems, that are common in such applications.

We are only interested in the estimation of the time-delay. Some methods also estimate the other parameters needed for a complete model of $G(p)$. We consider these other parameters as nuisance. When estimating the time-delay, the objective can be either of the following two:

1. Estimate the *best approximation time-delay*, i.e. the time-delay estimate that gives the "best" model approximation of the true system. What is "best" depends on the intended use of the model and can be measured in many

different ways. In automatic control, the time-delay estimate can be a means to achieve a good model in the frequency band relevant to the control [Lju02, FMS91], e.g. around the cross-over frequency. It is possible to choose the frequency band where to achieve a good model by designing the input signal spectrum and/or prefiltering the data [Lju99, Lju02]. In [Swa99] the *apparent time-delay* (the delay resulting from identifying a first order model with time-delay from the data) is used for control performance monitoring of PID control loops.

2. Estimate the *true time-delay* (T_d in Equation (1.1)). This is the case in "pure time-delay" estimation, diagnosis, radar range estimation [KQ92, Sum95], direction of arrival estimation with array antennas [HR97, Wik02, FHJ02], measuring blood velocity [LT99], averaging of measured signals [GS94] etc.

Since we have not decided on a special use of the time-delay estimate, we will in this thesis evaluate estimation methods according to the second objective.

We consider it as an advantage if a method can estimate time-delays that also consist of fractions of the sampling interval. However, some methods can only estimate time-delays that are a multiple of the sampling interval. Sometimes such methods can be used to initialize other more "free" methods.

Time-delay estimation has been studied in the literature for a long time, especially for pure time-delay systems [C$^+$81, Car87, Car93] but also for systems with dynamics [ÅH95, CHWW99, EDE89, FMS91, Hor00, Isa97, KG81, Lju02, NL91, Pup85, WZ01, Swa99, IHD01b, IHD00]. However, there is still no clear agreement on which method is "best" for systems with dynamics.

1.2 Purpose

The purpose of this thesis is to:

1. Review and classify existing time-delay estimation methods for dynamics systems according to their underlying principles.

2. Try to find the "best" time-delay estimation method for dynamic systems for different cases by comparing the quality of the estimates of methods using simulated data.

1.3 Contributions

The main contributions of this thesis are

1. Statement of a general time-delay estimation problem in linear systems encompassing both automatic control and signal processing applications (Equation 2.1).

2. Classification of existing time-delay estimation methods into classes with common principles and pointing out connections between the classes (Chapters 3-6).
3. Serious comparison of several time-delay estimation methods using confidence intervals and other methods on Monte Carlo simulated data to see which method is the the best and which method to use in which case (Sections 8-9).
4. Properties of classes of methods from simulations.
5. Some theoretical properties of local minima for explicit time-delay parameter methods using first order model with time-delay (Section 5.1.2).
6. Improvements and modifications of old and development of new time-delay estimation methods:
    (a) Zero guarding for making phase methods more robust (Section 4.2.7).
    (b) The prefiltered Arxstruc method `met1struc` (Section 5.2.3).
    (c) Exact time-delay from the sampling process (Section 5.3.3).
    (d) CUSUM thresholding on impulse and step response estimates (Section 4.1.6).
    (e) Area and moment methods on estimated step and impulse responses instead of measured ones (Section 6.1-6.2).
    (f) A cepstrum-like method for the separation of the time-delay from the dynamics of a linear time-delay system (Section 4.1.4).
7. Development of a MATLAB toolbox for managing and analyzing data from factorial experiments [Bjö]. Included in this is a method and a software for evaluating performance of parameter estimation methods with the aid of ANOVA and confidence intervals for pair-wise comparisons (Section 7.5).
8. Matlab implementation of some time-delay estimation methods (parts the methods in Section 7.2).

Much of the contents in this thesis is based on the reports [Bjö02, Bjö03a, Bjö03b, Bjö03d, Bjö03c, Bjö03f, Bjö03e, BL03], in which also more details can be found.

1.4 Outline

This thesis is structured as follows:

The next chapter, Chapter 2, presents several time-delay estimation (TDE) problems and properties of time-delay systems. Chapter 3 introduces a classification of TDE methods, while Chapters 4-6 describes the principles and properties of some classes of TDE methods. Several methods have been compared and investigated experimentally with Monte Carlo simulations. Chapter 7 lists these methods and describes the simulation setups and analyses. Next, in Chapter 8, method parameters are chosen and method properties are investigated using simulations. Chapter 9 contains a comparison of methods, also using simulations. After that, Chapter 10 contains a discussion and gives recommendations on the choice of method and gives conclusions about method properties and ends with a description of possible future work. Appendix A contains validation graphs for the analysis.

Part I

Time-delay estimation problems and methods

2

Time-delay Estimation Problems and Time-Delay Systems

The first section of this chapter states a general linear time-delay estimation (TDE) problem that encompasses most TDE problems found in the literature on automatic control and signal processing. Then some special cases of the general problem are listed, among others the problem from the introduction, namely to estimate T_d in

$$y(t) = G(p)u(t) + n(t) = G_r(p)u(t - T_d) + n(t)$$

(where $G_r(p)$ is a linear dynamic system without time-delay), which is the problem that is treated in this thesis. The second section lists some properties of linear dynamic systems with time-delay.

2.1 Time-Delay Estimation Problems

A general linear TDE problem is

$$\begin{aligned} y_1(t) &= G_1(p)u(t) + n_1(t) \\ y_2(t) &= G_2(p)u(t - T_d) + n_2(t) \end{aligned} \tag{2.1}$$

where, the signals $y_1(t)$ and $y_2(t)$ are measured, $n_1(t)$ and $n_2(t)$ are measurement noise and $G_1(p)$ and $G_2(p)$ are linear systems (without time-delay). The time-delay to be determined is T_d. The signals can be either wideband or narrowband. The signals can be either real valued or complex valued. Complex (or *analytic*) signal representation is often used for narrowband signals but can also be used for wideband signals. The impulse responses $g_1(t)$ and $g_2(t)$ of $G_1(p)$ and $G_2(p)$ can also be complex valued. Complex signals and impulse responses are commonly used for bandpass systems, e.g. in radar and communications.

Some special cases of the general problem (2.1) are:

1. With the noise $n_1(t) = 0$ and $G_1(p) = 1$ we have the *active time-delay estimation* (TDE) problem [MG$^+$98, Qua81] (we rename y_2 to y and n_2 to n):

$$y(t) = G(p)u(t - T_d) + n(t) \tag{2.2}$$

   This occurs in system identification, which is useful for automatic control and range estimation in radar etc. Special cases of active TDE are:

   (a) In, for example, radar with targets made of several scatterers or radio communications with multipath the following model could be appropriate:

$$y(t) = \sum_{m=1}^{M} g_m u(t - \tau_m) + n(t) \tag{2.3}$$

   The quantity M is the number of reflections in the target or multipath, g_m is the m^{th} reflection coefficient and τ_m is the time-delay to the m^{th} reflection. Here the single time-delay T_d seems to be replaced by one delay for each reflection. If we define $T_d = \min_m \tau_m$ and let the rest of the sum in (2.3) define a linear system G, we are back to problem (2.2).

   (b) Another special case of active TDE is when the system is under feedback, which is the case in automatic control. Then the input signal will be correlated with the output signal of previous times. This case is also called *closed loop*. In the same way the case without feedback is called *open-loop*.

   (c) $G(p) = \alpha$ with α being a constant. This problem occurs, for example, in radar with point targets [KQ92, Sum95].

   (d) $G(p) = 1$, which is a special case of 1c.

2. With the noise $n_1(t) \neq 0$ and $u(t)$ unknown we have the *passive time-delay estimation* problem [MG$^+$98, Qua81]. This case happens when a signal $u(t)$ has traveled two different paths and are measured with two sensors, e.g. in localization of radio sources by *Time Delay of Arrival* (TDOA) [HR97, FHJ02, Wik02] or beamforming of audio signals from an array of microphones in a car [Nyg03]. Special cases of passive TDE are:

   (a) $G_1(p) = 1$ and $G_2(p) = \alpha$ with α being a constant.

   (b) $G_1(p) = G_2(p) = 1$. This case is for example found when averaging several received signals with small time shifts, e.g. in ultrasonic imaging [GS94] and radar, with unknown arrival time [GS94] in order to increase the signal-to-noise ratio.

A *nominal active scenario* can be defined for the the active TDE problem [Car93, MG$^+$98]. In this scenario $n_2(t)$ is white Gaussian noise and $G_2 = 1$. The (asymptotically optimum) Maximum Likelihood estimate for this scenario is the matched filter which is the same as finding the highest maximum of the cross-correlation function $R_{y_2 u}(\tau)$ between $y_2(t)$ and $u(t)$ [MG$^+$98]. We will discuss this in Section 4.1.5.

Also for the passive TDE problem a *nominal passive scenario* can be defined [Car93, MG$^+$98]. In this scenario $G_1(p) = G_2(p) = 1$, $u(t)$ is a Gaussian random signal and $n_1(t)$ and $n_2(t)$ are mutually uncorrelated, zero mean, white Gaussian noises which are also uncorrelated with $u(t)$. The (asymptotically optimum) Maximum Likelihood estimate for this scenario is the generalized cross correlator [Car93], which consists of cross-correlation of prefiltered signals.

2.2 Time-Delay Systems

In this section we will state some properties of linear time-delay systems, i.e. systems of the form $\bar{G}(s) = \bar{G}_r(s) \cdot e^{-sT_d}$ (continuous-time) or $G(z) = G_r(z) \cdot z^{-d}$(discrete-time) where $\bar{G}_r(s)$ and $G_r(z)$ are transfer functions without time-delay.

We have the following properties for such systems:

- A pure time-delay $G(s) = e^{-sT_d}$ is a linear system.
- A pure time-delay $G(s) = e^{-sT_d}$ is an allpass system.
- A continuous-time time-delay system is of infinite dimension since an infinite number of values are needed to describe the state of the system at each point of time [MZJ87, p. 25-26].
- A continuous-time time-delay system can in state space form be described by a system of differential-difference equations [MZJ87, p. 26-27], i.e. combined differential and difference equations.
- The transfer function $G(s) = e^{-sT_d}$ of a continuous-time time-delay system is not a rational function of s. $G(s)$ has an infinite number of poles, which is consistent with the system's infinite dimensionality. See [MZJ87, p. 29].
- If the sampling period is constant and the delays are integral multiples of the sampling period, then a discrete-time time-delay system in state space form can be described by a system of pure difference equations. Such systems will be of finite dimension. See [MZJ87, p. 27-28] for more details.
- On the other hand, if the sampling period is not constant, then a discrete-time time-delay system cannot be described by pure difference equations. Differential-difference equations are needed [MZJ87, p. 28].

- The transfer function $G(z)$ of a discrete-time time-delay system is a rational function of z. $G(z)$ has a finite number of poles, which is consistent with the system's finite dimensionality. See [MZJ87, p. 32].

A hybrid, or mixed system, consists of a continuous-time part and a discrete-time part. An important example is a sampled continuous-time system. See [MZJ87, pp. 33] for more information.

3

Classes of Active Time-Delay Estimation Methods

Most methods that have been suggested for active time-delay estimation (Section 2.1) (both in control and signal processing) can be put into one of the following classes:

1. *Time-delay approximation methods.* The input and output signals are represented in a certain basis and the time-delay is estimated from an approximation of the relation (a model) between the signals in this basis. The time-delay is not an explicit parameter in the model. Depending on the basis there are several subclasses:

   (a) *Time domain approximation methods.* The basis consists of impulse functions $\delta(t - t_n)$. The time-delay is the delay for the impulse response to start (become nonzero) [Bjö03d, CHWW99, Isa97, KG81]. Finding the maximum of the cross-correlation between input and output, which is a common method [MG$^+$98, Car93], is in principle the same thing. Methods that over-parameterize the numerator of a transfer function model, e.g. [Kur79, KG81], also belong to this class.

   (b) *Frequency domain approximation methods.* The basis consists of complex sinusoids $e^{i\omega t}$. The time-delay is estimated from the phase of the time-delay $e^{-i\omega T_d}$ [IHD01b, Isa97, FHJ02, GS94, Bjö02, HR97]. A time-delay is a equivalent to a phase shift.

   (c) *Laguerre domain approximation methods.* The time-delay is estimated from a relation between the input and output signals expressed in continuous-time or discrete-time Laguerre functions [FM99b, Fis99].

Also other bases for the signals are possible, e.g. Kautz functions [Kau54, Wah94, SBS97]. There are two independent steps in these methods: 1) Estimate the approximation model. 2) Estimate the time-delay from the model.

2. *Explicit time-delay parameter methods.* The time-delay is an explicit parameter in the model.

   (a) *One-step explicit methods.* The time-delay and the other model parameters are estimated simultaneously. Two variants are possible:

      i. Model and estimate the time-delay as a continuous parameter in a continuous-time model. The time-delay is thus not restricted to be a multiple of the sampling interval but can be a subsample time-delay. See for example [NL91, Lju02].

      ii. Model and estimate the time-delay as a discrete parameter in a discrete-time model. Estimating several models, e.g. ARX models, with a complete set of time-delays and choosing the best is of this subclass ([Swa99, IHD00] and Section 5.2.3).

   (b) *Two-step explicit methods* [EDE89, Pup85]. Alternating between estimating the time-delay and the other parameters.

   (c) *Sampling methods.* Utilizing the sampling process to derive an expression for the time-delay. For example, zero-order-hold (zoh) sampling of a system with subsample time-delays creates an extra zero [FMS91].

3. *Area and moment methods* [ÅH95, Bjö03b, Ing03, WZ01]. These methods utilize relations between the time-delay and certain areas above or below the step response $s(t)$ and certain moments of the impulse response $h(t)$ (integrals of the type $\int t^n h(t)dt$). There are two independent steps: 1) Estimate or measure the step or impulse response. 2) Estimate the time-delay from these responses.

4. Higher-order statistics (HOS) methods. Their main advantage is that noise with a symmetric probability density function (PDF), e.g. Gaussian, theoretically can be removed completely by HOS [NM93]. If the desired signal has a symmetric PDF, it will disappear as well. In [NP88], bispectra and 3^{rd} order moments are used and methods in the 2D time and frequency domains, similar to subclasses 1a and 1b, are presented. They assume $G_r = 1$.

Methods for the passive TDE problem (Section 2.1) should also be possible to use for the active problem if the two noises $n_1(t)$ and $n_2(t)$ in Equation (2.1) are allowed to have different powers. Then $n_1(t)$ could be zero and we have the active TDE problem. Active TDE is thus a special case of passive TDE. The opposite, to use active TDE methods for passive TDE problems, could be possible if the power of the noise $n_1(t)$ is low.

4

Time-Delay Approximation Methods

In time-delay approximation methods, a model without an explicit parameter for the time-delay is estimated. From this model the time-delay is estimated. Depending on in which domain the input and output signals are represented this is performed in different ways. Section 4.1 describes the time domain, Section 4.2 the frequency domain and Section 4.3 the Laguerre domain.

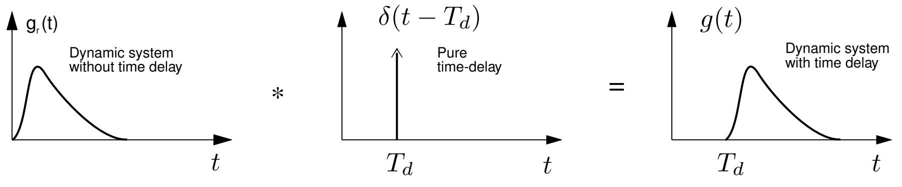
4.1 Time Domain Approximation Methods (Thresholding Methods)

4.1.1 Principles

In *time domain approximation methods* an approximation of the system including the time-delay is represented in the time domain. This means that the input, output and noise signals are represented in this domain, i.e. in the basis where the basis functions are impulse functions $\delta(t - t_n)$. The time-delay is estimated by measuring the time-delay to the start (the beginning of the nonzero part) of an estimated impulse response of the system. Since this can be done by thresholding these responses, these methods are here also called *thresholding methods*.

The impulse response of a system with time-delay is created as in Figure 4.1. To the left the impulse response $g_r(t)$ of the system without time-delay is depicted. In the middle the pure time-delay is seen as a system whose impulse response $\delta(t - T_d)$ is an impulse at the time equal to the time-delay T_d. To the right is the impulse response $g(t)$ of the system with time-delay, which is the convolution of the two systems to the left: $g(t) = g_r(t) * \delta(t - T_d)$.

Figure 4.1 The creation of the impulse response of a system with time-delay.



g_r(t)
Dynamic system without time delay
t
*
δ(t − T_d)
Pure time-delay
T_d
t
=
g(t)
Dynamic system with time delay
T_d
t

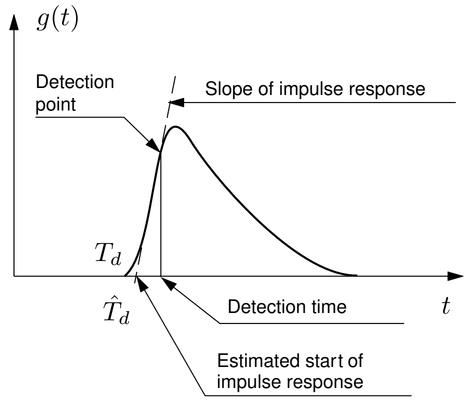
If an impulse response $\hat{g}(t)$ is available, directly measured or estimated, the time-delay can be estimated by either of the following two approaches:

1. Separation of time-delay and dynamics and then detection:
   (a) First, separate the pure time-delay $\delta(t - T_d)$ from the the dynamics $g_r(t)$ of the system $g(t)$. This is the reverse operation of the creation of the system in Figure 4.1 and results in an estimate $g_d(t)$ of the pure time-delay $\delta(t - T_d)$. See Section 4.1.4 for a simple example. This can be compared with the corresponding separation for frequency domain methods that will be presented in Sections 4.2.4-4.2.5.
   (b) Then, estimate the time when the maximum of the estimate $g_d(t)$ of the time-delay $\delta(t - T_d)$ occurs. This is the time-delay estimate $\hat{T}_d$.
2. Direct detection of the start of the impulse response ([KG81, Isa97, CHWW99] and Section 4.1.3):
   (a) Detect when the impulse response has started to rise (become nonzero).
   (b) Estimate the start time of the impulse response. An idea is to go backwards from the detection time depending on the slope of the impulse response (Figure 4.2).

In some signal processing applications where $G_r(s) = \alpha$, with α being a constant (case 1c in Section 2.1), the separation of $g(t)$ into $\delta(t - T_d)$ and $g_r(t)$ is not needed because the impulse response of the whole system with time-delay will be $g(t) = \alpha\delta(t - T_d)$, since $g_r(t) = \alpha\delta(t)$. The ubiquitous estimation of the cross-correlation between the input and output signals as a means to estimate time-delays [MG+98] is only a way to estimate the impulse response. See Section 4.1.5.

If we use the maximum of the impulse response as an estimate of the time-delay for systems with dynamics, i.e. $G_r(s)$ is a dynamic system and not only a constant, we would get a bias in the estimate. This is the motivation for the separation in estimation approach 1 above.

Figure 4.2 Estimation of time-delay by detecting the start of the impulse response.


$g(t)$
Detection point
Slope of impulse response
T_d
$\hat{T}_d$
Detection time
t
Estimated start of impulse response

4.1.2 Estimating the approximation model

Before we can estimate the time-delay from the impulse response as in Section 4.1.1, we need to estimate the impulse response from input output data. This is the topic of this section.

The impulse response $g(t)$ can be expressed as

$$g(\tau) = \frac{R_{yu}(\tau)}{\lambda} \tag{4.1}$$

when the input signal is white noise with autocorrelation function $R_u(\tau) = \mathrm{E}\{u(t+\tau)u(t)\} = \lambda\delta(\tau)$ [Lju99, p. 170]. The quantity λ is the power of the input signal. Estimates of the cross-correlation function $R_{yu}(\tau) = \mathrm{E}\{y(t+\tau)u(t)\}$ and the input signal power λ can be obtained by:

$$\hat{R}_{yu}(\tau) = \frac{1}{N}\sum_{t=\tau+1}^{N} y(t)u(t-\tau)\,, \qquad \hat{\lambda} = \frac{1}{N}\sum_{t=1}^{N} u(t)^2 \tag{4.2}$$

If the input is white, an estimate of the impulse response is then

$$\hat{g}(\tau) = \frac{\hat{R}_{yu}(\tau)}{\hat{\lambda}}. \tag{4.3}$$

This is called correlation analysis [Lju99].

A different way of estimating the impulse response is to estimate a parametric FIR model of the system with the Prediction Error Method (PEM) (Section 4.1.5). The model will be a linear regression and the estimate is given by a least-squares estimate:

$$\hat{\theta}_N = \underbrace{\left[\frac{1}{N} \sum_t \varphi(t)\varphi(t)^T \right]^{-1}}_{A} \underbrace{\left[\frac{1}{N} \sum_t \varphi(t)y(t) \right]}_{B} \tag{4.4}$$

with $\varphi(t) = [u(t), u(t-1), ...]$ [Lju99, p. 204]. This is essentially the same as Equation (4.3). If the input signal $u(t)$ is white, then the matrix A in Equation (4.4) will be $A = \hat{\lambda} I$ and the matrix B will be

$$B = \begin{bmatrix} \hat{R}_{yu}(0) \\ \hat{R}_{yu}(1) \\ \vdots \end{bmatrix} \tag{4.5}$$

and $\hat{\theta}_N = \begin{bmatrix} \hat{g}_0 & \hat{g}_1 & \dots \end{bmatrix}$ is the same as $\hat{g}(\tau) = \begin{bmatrix} \hat{g}_0 & \hat{g}_1 & \dots \end{bmatrix} = \begin{bmatrix} \hat{g}(0) & \hat{g}(1) & \dots \end{bmatrix}$ in Equation (4.3).

The estimate $\hat{\theta}_N$ will be Gaussian distributed when the noise is Gaussian. Even if the noise is not Gaussian, $\hat{\theta}_N$ will often be asymptotic Gaussian when $N \to \infty$ [Lju99, p. 556].

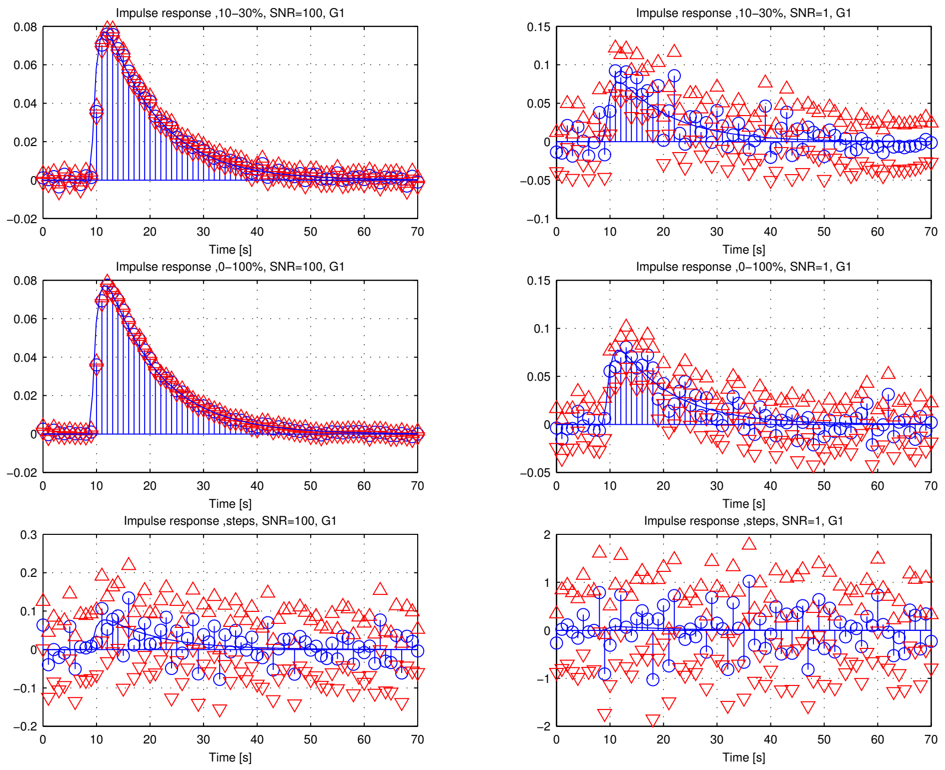
Figure 4.3 displays some estimated impulse responses for the system $G_1(s)$ in Section 7.3 for the three input signals in the standard benchmark (Section 7.3.1) and the two SNRs 100 and 1. The estimation is performed according to Equation 4.4 using the arx command in the MATLAB SYSTEM IDENTIFICATION TOOLBOX. We see that in some cases the estimate is very inaccurate. This makes the time-delay estimation a hard job, both for the thresholding methods in Sections 4.1.3 and 4.1.6 and the area and moment methods in Sections 6.1 and 6.2. In reference [CHWW99] a two-step improvement method to reduce the uncertainty of the impulse response is suggested (see Section 4.1.3).

When we have an estimate of the impulse response, we can get an estimate of the step response by simulating the estimated system (represented by its impulse response) with a step as input signal. This is the same as integrating the impulse response. The numerical integration of the impulse response estimate can be done in different ways. By the integration, which is an averaging operation, we can hope that the uncertainty of the impulse response estimate (Figure 4.3) is reduced. If the impulse response coefficients to integrate are Gaussian distributed then the step response will also be Gaussian since a sum of Gaussian random variables is Gaussian.

In [KG81], ARMAX models are used instead of FIR models (=impulse response). Parts of the dynamics of the system are then modeled by the denominator polynomial and the numerator polynomial is not equal to the impulse response but the numerator still contains the time-delay. When estimating the time-delay, the numerator can be treated as the impulse response is treated in Section 4.1.1 and 4.1.3.

Figure 4.3 Impulse response estimate of the system G_1 by the MATLAB function arx for different input signal types $\updownarrow$ and different SNRs $\leftrightarrow$(Section 7.3). No prewhitening of the input signal (Section 7.3). The solid line is the true impulse response. The circles are the estimated impulse response and the triangles mark $\pm$two estimated standard deviations. Note the different ranges of the vertical axes.

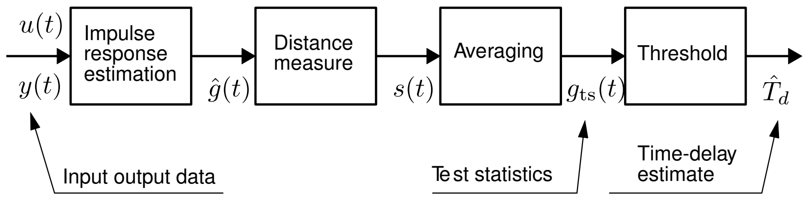
(t130f1 SNR out))



4.1.3 Estimating the start of the impulse response

Most time domain methods for dynamic systems with time-delay in the literature detect the start of (the start of the nonzero part of) the impulse response (approach 2 in Section 4.1.1). In this section we will discuss how this can be done. Figure 4.4 uses the same structure and terminology as in [Gus00] to describe the time-delay estimation process. After the impulse response has been estimated, $\hat{g}(t)$, we form a distance measure $s(t)$, which will ideally become nonzero when the impulse response has started to rise. For us $s(t)$ is simply the estimated impulse response $\hat{g}(t)$. Then we perform an averaging operation to reduce the noise. This results in the test statistics $g_{\text{ts}}(t)$, which then is thresholded to detect the start of the impulse response. Finally, the time-delay estimate is given by an estimate of the change time of the impulse response (see Figure 4.2).

Figure 4.4 Steps in time-delay estimation in the time domain.



The averaging to get the test statistics can be accomplished by one or several of:

Step response. Integration to step response.

Cumulative sum (CUSUM). Perform an averaging of the estimated impulse or step response by CUSUM [Pag54, Gus00, GLM01] before the thresholding. There are two user-selected parameters, the drift $\nu(t)$ and the threshold $h(t)$. See Algorithm 1. See Figure 4.6 for an example of CUSUM thresholding of an impulse response estimate.

Carlemalm impulse response. In [CHWW99] a special technique is used to decrease the uncertainty of the estimated impulse response, cf. Figure 4.3. First, all coefficients of the impulse response are estimated recursively by an LMS filter [Gus00, GLM01]. Then these estimates are improved by one Kalman filter [Gus00, GLM01] for each coefficient. The estimated coefficients from the LMS filter are used as measurement signals for the Kalman filters. The results from the Kalman filters are the second estimates of the coefficients (the first estimates come from the LMS filter) and estimates of their covariance. This technique requires white input and Gaussian noise.

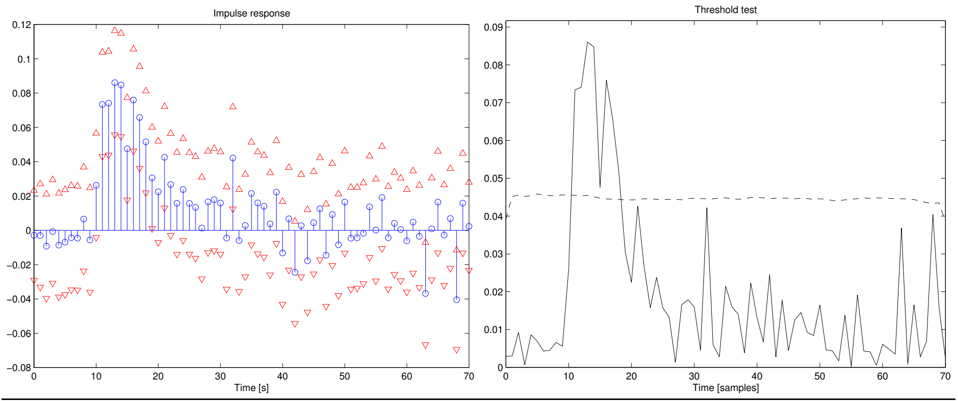
Direct. Use the estimated estimated impulse or step response directly for thresholding. Using the impulse response directly is of course no averaging. See Figure 4.5 for an example of direct thresholding of a an impulse response estimate.

The thresholding (Figure 4.4) can be performed by:

Fixed threshold. The threshold is fixed and data independent.

Relative threshold. The threshold depends on the uncertainty of the impulse or step response estimate. If we let the threshold be proportional to the uncertainty of the impulse or step response estimate we will get a specified risk for false alarm (saying that the time-delay is over when it is not). By lowering this risk (increasing the threshold) the probability of detection will also be lowered.

Figure 4.5 Direct thresholding of impulse response. Left: Estimated impulse response with uncertainty . Right: Estimated impulse response and threshold. Simulated input signal of type RBS 10-30% and system G_1 (Section 7.3). SNR=1. The estimated time delay with idimp4 (Section 7.2) ($h_{\text{std}} = 3$) was $\hat{T}_d = 11$. True time-delay $T_d = 10$. (t134d1)



Algorithm 1 CUSUM detector.

Design parameters: Drift $\nu(t)$ and threshold$h(t)$ that may be time dependent.

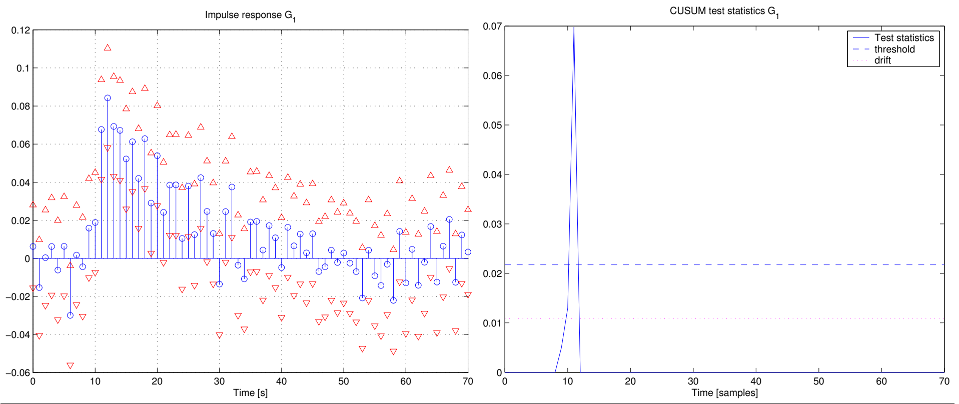
Output: Detection time t_a and perhaps the estimate of the change time $\hat{k}$.

Input: Distance measure $s(t)$.

Internal variable: Test statistics $g_{\text{ts}}(t)$.

```
1. $t = 0$, $g_{\text{ts}}(-1) = 0$
2. $g_{\text{ts}}(t) = g_{\text{ts}}(t-1) + s(t) - \nu(t)$
3. If $g_{\text{ts}}(t) < 0$ then $g_{\text{ts}}(t) = 0$, $\hat{k} = t$
4. If $g_{\text{ts}}(t) > h(t) > 0$ then $t_a = t$, goto 6
5. $t = t + 1$ , goto 2
6. The detection time is $t_a$. An estimate of the change time is $\hat{k}$.
```

Figure 4.6 CUSUM thresholding of impulse response. Left: Impulse response with uncertainty . Right: Test statistics $g(t)$, threshold h and drift ν for CUSUM on impulse response. Simulated input signal of type RBS 10-30% and system G_1 (Section 7.3). SNR was 1. The estimated time delay with idimpCusum3 ($h_{\text{std}} = 2$ and $\nu_{\text{std}} = 1$) was $\hat{T}_d = 11$. True time-delay $T_d = 10$. (t146b1)



Max threshold for detection. Using a “max threshold for detection” is the same as finding the maximum of the cross-correlation function. This is a suitable approach when the output signal is a pure time-delay of the input signal but not if there also is a dynamic system between the input and output. See Section 4.1.1.

Constant False Alarm Ratio (CFAR). With CFAR the threshold is selected to give a constant false alarm rate by looking at the values of the estimated cross-correlation function (impulse response) close to the time-delay that is being tested [MG+98, NS95]. These close-by values are used in an estimate of the noise level. Compare with the Section “Relative threshold” above. This method is common in radar.

Fault detection approach. In [Isa97] a time-delay estimation method is presented which is based on fault detection. It assumes the input signal to be a step but as we know that a step response can be estimated for many input signals the method is more general than stated in [Isa97]. In the method, for each sample of the step response a Kalman filter is started which tries to track the output. This will give a bank of Kalman filters. The filter that best tracks the output is detected and gives via its starting time the time-delay estimate.

Carlemalm relative threshold. In [CHWW99] a thresholding is described which is similar to “relative threshold” above but simultaneously takes into account more than one of the estimated impulse response coefficients.

Kurz detection. In [Kur79, KG81] the time-delay is estimated by detecting when the numerator coefficients of an estimated linear model no longer are zero. Due to noise this is not easy and a special technique is used.

When a change in the estimated impulse response has been detected, the change time can be estimated by going backwards from the detection time depending on the slope of the impulse response (Figure 4.2). See [Gus00] for estimating the change time when using CUSUM. In the simulations of thresholding methods in this work the detection time and not an estimated change time was used as the time-delay estimate. For example, in the CUSUM detection in Algorithm 1 the detection time t_a was uses as the time-delay estimate. This should cause bias as is discussed in Section 8.1.2.

4.1.4 Separating the time-delay and the dynamics

In this section we will give an example on how the dynamics can be separated from the time-delay (approach 1 in Section 4.1.1). We start with an impulse response of a system with time-delay $g(t) = g_r(t) * \delta(t - T_d)$ as in Section 4.1.1. See Figure 4.7 (top left) for an example. The Fourier transform of $g(t)$ is $G(i\omega) = G_r(i\omega) \cdot e^{-i\omega T_d}$. Then by taking the real part of $G(i\omega)$ we get

$$G_{\text{fr}}(i\omega) = \operatorname{Re}\left\{G(i\omega)\right\} = \operatorname{Re}\left\{G_r(i\omega) \cdot e^{i\omega T_d}\right\} = A_{\text{fr}}(i\omega)\sin\left(T_d\omega + \varphi(i\omega)\right),$$

where

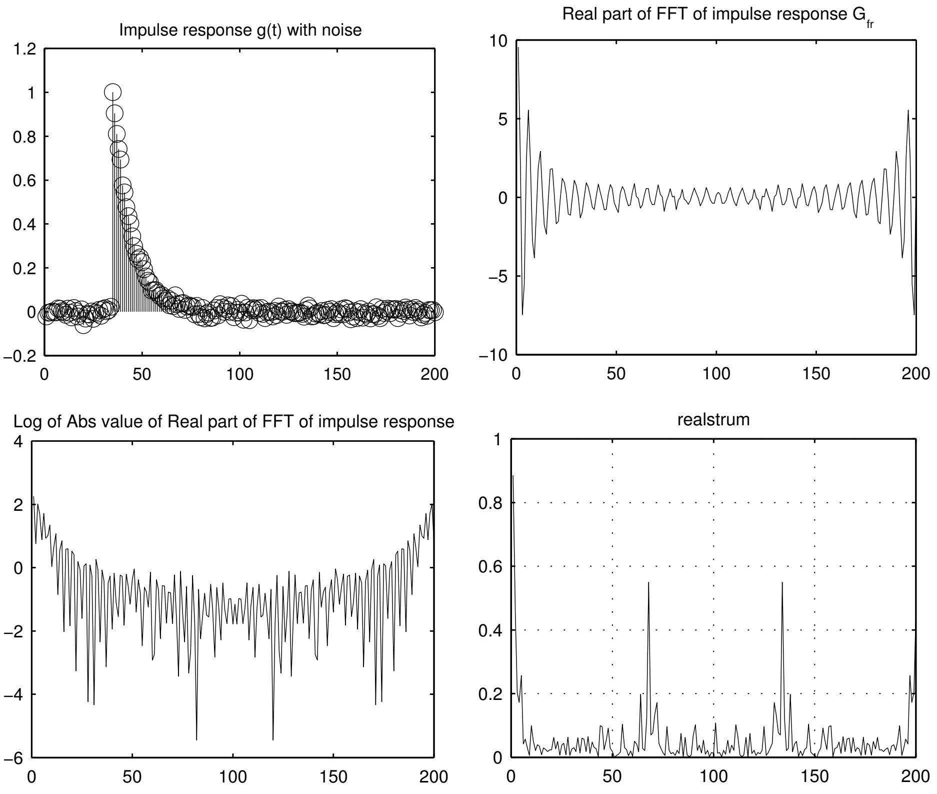
$$\begin{aligned} A_{\text{fr}}(i\omega) &= \sqrt{\left(\operatorname{Re} G_r(i\omega)\right)^2 + \left(\operatorname{Im} G_r(i\omega)\right)^2} \\ \varphi(i\omega) &= \arcsin\left(\frac{\operatorname{Re} G_r(i\omega)}{A_{\text{fr}}(i\omega)}\right). \end{aligned}$$

The function $G_{\text{fr}}(i\omega)$ consists of $A_{\text{fr}}(i\omega)$ which is modulated with a sinusoid with "frequency" T_d and phase shift $\varphi(i\omega)$ (Figure 4.7 (top right)). One idea to estimate the "frequency" would be to study the "spectrum" of $G_{\text{fr}}(i\omega)$. Since we already are in the frequency domain, we could compute the inverse Fourier transform and look for the peak at "frequency" T_d. We see in Figure 4.7 (top right) that there is an oscillation in $G_{\text{fr}}(i\omega)$ but its amplitude is small. To increase the effect of the oscillation we take the logarithm (after taking the absolute value to avoid the logarithm of negative values) which can be used to amplify weak parts of a positive signal (Figure 4.7 (bottom left)). Finally, we compute the inverse Fourier transform and look for peaks at the "frequency" $2T_d$. Because the absolute value doubles the base frequency of the periodic signal (the sinusoid) we need the number 2.

This method is similar to using *cepstrum*, in which we only take the absolute value instead of the absolute value of the real part as above, to find the time-delay of an echo [DC02, GLM01] (problems 1c and 2a in Section 2.1).

When a change in the estimated impulse response has been detected, the change time can be estimated by taking the maximum of an interpolation of the estimated

Figure 4.7 An example of separation of the dynamics from the pure time delay. The true time delay is $d_0 = 33$ samples . Top left is an impulse response $g(t)$ of the system $G(z) = \frac{1}{(z-0.9)} \cdot z^{-33}$ with noise (SNR=100). Top right is the real part of the Fourier transform of the impulse response, G_{fr}. Bottom left is the logarithm of the absolute value of the top right signal. The last graph (bottom right) shows the inverse Fourier transform of the bottom left signal and has a peak at the double time-delay $(2(d+1))$. (t237a)



approximation $g_d(t)$ of the pure time-delay $\delta(t - T_d)$. This can achieve subsample time-delay estimates [Mod91, LT99].

If the input signal is not white, and especially when it is oscillatory (there are dominating frequencies), then the cross-correlation function (the impulse response) will have several local maxima. See [Pup85, JD93].

4.1.5 Relation between PEM, cross-correlation, matched filter and maximum likelihood estimation

In this section we will discuss the relation between the Prediction Error Method (PEM) [Lju99], cross-correlation method, matched filter [Hän91] and maximum likelihood estimation of the time-delay. The relation between PEM and correlation analysis was shown in section 4.1.1. The relation between time-delay estimation in the time and frequency domains is discussed in Section 4.2.2.

The prediction error method (PEM) is a fundamental approach to estimate models of dynamic systems. In PEM the estimate $\hat{\theta}_N$ of the model parameter vector θ using N input output data is given (with a quadratic criterion) by

$$\hat{\theta}_N = \arg\min_{\theta} \frac{1}{N} \sum_{t=1}^{N} \frac{1}{2} \varepsilon^2(t|\theta),$$

where $\varepsilon^2(t|\theta) = y(t) - \hat{y}(t|\theta)$ is the prediction error of the model and $\hat{y}(t|\theta)$ is the one-step prediction of the model output [Lju99]. If the data are generated by linear systems and the model is linear, then $\hat{\theta}_N$ will converge to

$$\theta^* = \arg\min_{\theta} \bar{\mathrm{E}} \left\{ \frac{1}{2} \varepsilon^2(t|\theta) \right\}$$

as the number of data $N \to \infty$ [Lju99, Th.8.2, p.254]. The symbol $\bar{\mathrm{E}}$ means ensemble averaging for stationary stochastic processes (statistical expectation) and time averaging (as $N \to \infty$) for deterministic signals. For all details of this result see [Lju99].

In the cross-correlation method for time-delay estimation, the original signal $u(t)$ and the time-delayed signal $y(t)$ are compared. They are put close to each other. Then they are time-shifted until they agree the most. For stochastic processes this can more formally be written

$$\hat{d} = \arg\max_{\tau} \mathrm{E} \left\{ y(t) u(t-\tau) \right\}.$$

In reality this has to be implemented as

$$\hat{d} = \arg\max_{\tau} \sum_{t} y(t) u(t-\tau). \tag{4.6}$$

The purpose of a matched filter [Hän91] is to detect a known signal $u(t)$. This is done by sending the received signal $y(t)$ through a linear filter with the impulse response $h(t) = u(-t)$. The filter is matched to the signal $u(t)$. The output $z(\tau)$ (we here use τ as the time variable instead of t) of the filter is given by the well-known convolution sum

$$z(\tau) = \sum_{t} h(t-\tau) y(t) = \sum_{\tau} y(t) u(\tau - t).$$

The arrival time of the known signal can be estimated with the time when the filter output has its maximum value:

$$\hat{d} = \arg\max_{\tau} z(\tau) = \arg\max_{\tau} \sum_{\tau} y(t)u(\tau - t). \tag{4.7}$$

As is seen, this is the same as the cross-correlation method for time-delay estimation in Equation (4.6).

We will now show the relation between the PEM and the cross-correlation method (matched filter) for a pure time-delay. We will show that for the active TDE problem 1d in Section 2.1,

$$y(t) = u(t-d) + n(t) \text{ with } n(t) \text{ white noise,}$$

the prediction error method (PEM) in system identification and the matched filter are equivalent.

The PEM estimate uses the the one-step predictor $\hat{y}(t|\tau) = u(t-\tau)$ [Lju99] of the output signal $y(t)$ for a pure time-delay of size τ. The PEM estimate is [Lju99]

$$\begin{aligned} \widehat{d} &= \arg\min_{\tau} \sum (\hat{y}(t|\tau) - y(t))^2 = \arg\min_{\tau} \sum (u(t-\tau) - y(t))^2 \\ &= \arg\min_{\tau} \left[\sum u^2(t-\tau) + \sum y^2(t) - 2\sum y(t)u(t-\tau) \right] \\ &= \arg\min_{\tau} \left[-2\sum y(t)u(t-\tau) \right] = \arg\max_{\tau} \sum y(t)u(t-\tau), \end{aligned} \tag{4.8}$$

which is the same as the matched filter in Equation (4.7). The fourth equality sign in (4.8) follows since $\sum y^2(t)$ does not contain τ and $\sum u^2(t-\tau) = \sum u^2(t)$ does not depend on τ.

In [JD93, p. 289] it is asserted that there is equivalence between maximum likelihood estimation of the time-delay and a matched filter (cross-correlation) if the following assumptions are met:

- The active TDE problem 1d in Section 2.1 with discrete-time signals and time-delay d:

  $$y(t) = u(t-d) + n(t)$$

- The noise $n(t)$ is Gaussian with rational spectrum with only poles but it is not necessarily white.
- The duration of the input signal $u(t)$ is short compared to the length of the observation interval.
- The edge effects due to the finite observation interval are ignored.

The matched filter estimate is

$$\widehat{d} = \arg\max_{\tau} \sum_{t=0}^{L-1} y(t)u(t-\tau) \tag{4.9}$$

Algorithm 2 Direct and CUSUM thresholding of impulse or step response with relative threshold.

1. Choose the the relative threshold h_{std} and relative drift ν_{std}.
2. Estimate the impulse response and estimate the uncertainty of it.
3. Optionally, integrate to step response.
4. Thresholding. If the estimated impulse or step response is larger than a threshold then we consider the impulse (step) response to have started and this point of time is the time-delay estimate. The thresholding can be either
   - Direct thresholding (Section 4.1.3). The thresholds (different for each time) are $h(t) = h_{\text{std}} \cdot \hat{y}_{\text{std}}(t)$ and $\hat{y}_{\text{std}}(t)$ is the estimated standard deviation of the impulse or step response, respectively.
   - Cumulative sum (CUSUM) thresholding (Algorithm 1 and Section 4.1.3). The used drift and threshold are then $\nu(t) = \nu_{\text{std}} \cdot \hat{y}_{\text{std}}(t)$ and $h(t) = h_{\text{std}} \cdot \hat{y}_{\text{std}}(t)$.

for white noise (compare with Equations 4.7, 4.8) and

$$\widehat{d} = \arg\max_{\tau} \left[\mathbf{y}^T R_{nn}^{-1} \mathbf{u}(t-\tau)\right] \tag{4.10}$$

for colored noise. The matrix R_{nn} is the covariance matrix of the noise. The vector $\mathbf{y} = [y(0), \dots, y(L-1)]^T$. The vector $\mathbf{u}(\Delta t) = [u(0-d), \dots, u(L-1-d)]^T$.

With the above assumptions this is the same as the maximum likelihood estimate of the time-delay according to [JD93, p.289].

4.1.6 Some methods

Here some complete methods, i.e. with both estimation of the approximation model (Section 4.1.1) and estimation of the time-delay from the model (Sections 4.1.1 and 4.1.3), are listed.

Direct and CUSUM thresholding. In these methods the impulse or step response estimate is thresholded directly or with CUSUM with a relative threshold (Section 4.1.3). See Algorithm 2. In this thesis the detection time t_a was used as the time-delay estimate (cf. Algorithm 1) . The uncertainty $\hat{y}_{\text{std}}(t) = \hat{y}_{\text{std}}(0)$ was used for all times for CUSUM thresholding.

Kurz method. In [Kur79, KG81] a method for time-delay estimation is described. It assumes that the true system without the time-delay can be described by an ARMAX system structure [Isa97]. The model is of ARMAX structure with an extended numerator. The time-delay is estimated by detecting when the numerator coefficients no longer are zero. Due to noise this is not easy and a special technique is used [KG81, Isa97]. See also Kurz detection in Section 4.1.3.

Isaksson Fault detection approach. This method uses Fault detection approach in Section 4.1.3 on measured step responses. According to [Isa97], advantages are

- No "preliminary knowledge" of the system is needed.
- No large difference in performance between low and high SNR.
- The method is robust.
- The method usually gives accurate estimates of the time-delay.

Carlemalm. This is a combination of Carlemalm impulse response and relative threshold or Carlemalm relative threshold in Section 4.1.3.

4.2 Frequency Domain Approximation Methods (Phase Methods)

4.2.1 Time-delay in continuous-time

In *frequency domain approximation methods* an approximation of the system including the time-delay is represented in the frequency domain. The input, output and noise signals are represented in the frequency domain, i.e. in the basis where the basis functions are complex sinusoids $e^{i\omega t}$. As is well-known, a time-delay $e^{-i\omega T_d}$ is the same as a phase shift $-\omega T_d$ in the frequency domain. In these methods we estimate the slope of the phase of the cross-spectrum in the frequency domain. Therefore they are here also called *phase methods*.

If $\bar{G}(s)$ is a linear continuous-time system then

$$\frac{d}{d\omega} \arg \bar{G}(i\omega)\bigg|_{\omega=0} = 0$$

if the system do not contain a time-delay and

$$\frac{d}{d\omega} \arg \bar{G}(i\omega)\bigg|_{\omega=0} = -T_d$$

if the system contains a time-delay T_d.

4.2.2 Time-delay in discrete-time

Since we have sampled data, we would like to use the sampled frequency function $G(e^{i\omega T})$ instead of the continuous-time frequency function $\bar{G}(i\omega)$. This will work, since for low frequencies, $G(e^{i\omega T})$ does not differ much from the true frequency function $\bar{G}(i\omega)$ [LG91, p. 74]. A rule of thumb is that the agreement is good for frequencies up to 1/10 of the sampling frequency. The sampled frequency function $G(e^{i\omega T})$ can be estimated by nonparametric methods like spectral analysis or by parametric methods with arbitrary linear model structures [Lju99, LG91].

In time-delay estimation by spectral analysis, we utilize the formula [GLM01] $\Phi_{yu}(\omega) = G(e^{i\omega T})\Phi_u(\omega)$ or

$$G(e^{i\omega T}) = \frac{\Phi_{yu}(\omega)}{\Phi_u(\omega)}, \tag{4.11}$$

where the sampled frequency function $G(e^{i\omega T})$ is connected with the cross-spectrum $\Phi_{yu}(\omega)$ between output $y(t)$ and input $u(t)$ and the (auto) spectrum $\Phi_u(\omega)$ of the input. A natural estimate of $G(e^{i\omega T})$ is then to use the estimated cross-spectrum $\hat{\Phi}_{yu}(\omega)$ divided by the estimated (auto) spectrum $\hat{\Phi}_u(\omega)$:

$$\hat{G}(e^{i\omega T}) = \frac{\hat{\Phi}_{yu}(\omega)}{\hat{\Phi}_u(\omega)}. \tag{4.12}$$

The cross-spectrum and the (auto) spectrum can be estimated in different ways. Normally, the start is the periodogram [GLM01, Lju99]. Then it is smoothed with some method [GLM01, Lju99, Wik02].

In [HR95, HR97, Wik02] the cross-spectrum is then noise reduced by windowing in the time domain before returning to the frequency domain. This is possible for wideband signals whose cross-correlation function in the time domain is narrow.

Then the phase of $\hat{G}(e^{i\omega T})$ is studied to give an estimate of the time-delay

$$\hat{T}_d = -\frac{d}{d\omega} \arg \hat{G}(e^{i\omega T})\bigg|_{\omega=0}$$

where the derivative is approximated in a suitable way. The reference [Hor00, IHD01b] contains an approximation of the derivative for the active TDE problem (2.2). The derivative is easier to approximate for the passive TDE problem 2a (Section 2.1) because then the time-delay is the only thing that affects the phase. See [Wik02] for an example of approximation of the derivative in this case.

Time-delay estimation by spectral analysis is equivalent to time-delay estimation by cross-correlation analysis in the frequency domain. Since the cross-spectrum $\Phi_{yu}(\omega)$ is the discrete-time Fourier transform of the cross-correlation function $R_{yu}(\tau)$ [GLM01, p. 54],

$$\Phi_{yu}(\omega) = T_s \sum_{k=-\infty}^{\infty} R_{yu}(k) e^{i\omega k T_s},$$

and in the same way the autospectrum the Fourier transform of the autocorrelation function $R_u(\tau)$, we see that Equation (4.12) is the same as Equation (4.1) in the frequency domain (if the input signal is white). In the time domain (Equation (4.1)) a time-delay in the system will be a time-delay in $h(\tau)$. In the frequency domain (Equation (4.12)) a time-delay in the system will be a phase shift in $\hat{G}(e^{i\omega T})$. For an example of time-delay estimation by nonparametric methods see [Wik02].

If the model structure in a parametric method is a linear regression then the model can be estimated by cross- and autocorrelations. Compare with the FIR model in Section 4.1.2. For some examples of time-delay estimation by parametric methods see [Hor00, IHD01b, Bjö02].

4.2.3 Estimating the approximation model

To use the estimation methods in the previous section we first must estimate a discrete-time model of the true system with time-delay. One interesting model structure to use is the Laguerre model structure. In this section we briefly describe this structure.

If a discrete-time system $G(z)$ is is strictly proper, asymptotically stable and continuous in $|z| \geq 1$, then it can be written [Wah91, p. 552]:

$$G(z) = \sum_{k=1}^{\infty} d_k \frac{\sqrt{(1-\alpha^2)T_s}}{z-\alpha} \left(\frac{1-\alpha z}{z-\alpha} \right)^{k-1} = \sum_{k=1}^{\infty} d_k L_k(z) \tag{4.13}$$

with $|z| \geq 1$ and $-1 < \alpha < 1$. T_s is the sampling interval and d_k are some coefficients. We change from the Z-transform variable z to the delay operator q and write the functions $L_k(z)$ as

$$L_k(q) = \frac{\sqrt{(1-\alpha^2)T_s}q^{-1}}{1-\alpha q^{-1}} \left(\frac{-\alpha + q^{-1}}{1-\alpha q^{-1}} \right)^{k-1}. \tag{4.14}$$

These functions are called the discrete-time Laguerre functions in the frequency domain.

A model with a finite number of Laguerre functions looks like

$$y(t) = \hat{G}(q)u(t) + v(t) \tag{4.15}$$

$$\hat{G}(q) = \sum_{k=1}^{n_{\text{lag}}} d_k L_k(q) \tag{4.16}$$

with $y(t)$, $u(t)$ and $v(t)$ being the output, input and noise signals respectively. The model $\hat{G}(q)$ is an approximation of the true system $G(z)$ in Equation (4.13).

In [Hor00, IHD01b] a Laguerre model was used. There is, however, nothing that prevents us from using a model of a different structure and estimate the time-delay from it. For example, FIR, ARX or output error (OE) model structures [Lju99] could be used. We will see examples on this later.

4.2.4 Continuous-time estimation using Padé approximation

A time-delay e^{-sT_d} can be approximated by a Padé approximation of the first order (see for example [Mat96, GLM01, Isa97, p. 149])

$$e^{-sT_d} \approx \frac{1 - sT_d/2}{1 + sT_d/2} = H_1(s) \tag{4.17}$$

for small $|sT_d|$.

The amplitude of both the time-delay $e^{-i\omega T_d}$ and the all-pass system $H_1(s)$ is equal to one. It is the phase that characterizes the time-delay and it is desirable that the all-pass system approximates this well. Padé approximations of time-delays of higher orders are also possible [Lam93, eq. (2.1) and (2.2)], [Isa97] or in [GVL96, p.572]. All zeros in a Padé approximation are non-minimum-phase [Isa97].

An approach to estimate the time-delay from a Padé approximation was described in [Isa97]. After the discrete-time Laguerre model has been estimated, its zeros are converted to continuous-time by means of

$$s_i \approx \frac{1}{T_s} \log(z_i) \tag{4.18}$$

This approximation is valid when the sampling period $T_s \to 0$ and the order n of the numerator polynomial and the order m of the denominator polynomial of the continuous-time system fulfills $n+1=m$ [ÅSH84]. .

Let the true continuous-time system be $\bar{G}(s) = \bar{G}_1(s) \cdot e^{-sT_d} = \bar{G}_1(s) \cdot \bar{G}_{ap}(s)$. We assume that all non-minimum-phase zeros originate from the time-delay and not from the remaining system $\bar{G}_1(s)$. The system $\bar{G}_1(s)$ is thus assumed to be minimum-phase. By comparing the non-minimum-phase zeros with the zeros of an Pade ́ approximation, the time-delay can be decided from [Isa97] as

$$\frac{\hat{T}_d}{2} = \sum_{RHP} \frac{1}{s_i}, \tag{4.19}$$

where RHP means Right Half Plane.

The time-delay estimate in number of sampling intervals is [Isa97, Hor00]

$$\hat{k} = \frac{\hat{T}_d}{T_s} + 1 \tag{4.20}$$

The number 1 in Equation (4.20) is added to give the time-delay after sampling since zero-order-hold sampling gives an extra time-delay of one sampling interval.

A problem with this method is that it can deliver complex valued time-delay estimates.

4.2.5 Using the phase of the discrete-time allpass part (DAP methods)

Assume the true continuous-time system is $\bar{G}(s) = \bar{G}_1(s) \cdot e^{-sT_d} = \bar{G}_1(s) \cdot \bar{G}_{ap}(s)$. The system $\bar{G}_1(s)$ is linear and time invariant. $\bar{G}_{ap}(s)$ is the time-delay. We estimate a discrete-time rational linear model $G(z)$ of $\bar{G}(s)$ and factorize $G(z)$ into a minimum-phase system $G_1(z)$ and an allpass system $G_{ap}(z)$: $G(z) = G_1(z)G_{ap}(z)$. (Allpass means $\left|G_{ap}(e^{i\omega})\right| \equiv 1$) We then consider $G_{ap}(e^{i\omega T_s})$ as an approximation of the time-delay $\bar{G}_{ap}(i\omega) = e^{-i\omega T_d}$ [Hor00, IHD01b]. A pure time-delay is of course an allpass system. We get the allpass part $G_{\mathrm{ap}}(z)$ of $G(z)$ if we put all non-minimum phase (outside the unit circle) zeros of $G(z)$ into $G_{\mathrm{ap}}(z)$ and add poles to $G_{\mathrm{ap}}(z)$ which are the non-minimum phase zeros mirrored in the unit circle.

If we approximate a continuous-time linear system $\bar{G}_{ap}(s)$ with a discrete-time linear system $G_{ap}(z)$ by sampling, then the frequency functions $\bar{G}_{ap}(i\omega)$ and $G_{ap}(e^{i\omega T_s})$ will agree well for low frequencies [GL97, p. 115-116]. [GL97] has the rule-of-thumb that frequencies up to 1/10 of the sampling frequency gives a good agreement.

From the allpass model $G_{ap}(e^{i\omega T_s})$ the time-delay can be estimated by taking the derivative of the phase $\varphi(\omega) = \arg G_{ap}(e^{i\omega T_s})$. The motivation for this is that the time-delay of the true system is given in this way:

$$\frac{d\bar{\varphi}}{d\omega} = \frac{d \arg \bar{G}_{ap}(i\omega)}{d\omega} = \frac{d}{d\omega} \arg e^{-i\omega T_{d=}} \frac{d}{d\omega} \{-\omega T_d\} = -T_d \tag{4.21}$$

Our time-delay estimate $\hat{T}_d$ is given by an approximation of the derivative of the phase $\varphi(\omega T_s) = \arg G_{ap}(e^{i\omega T_s})$:

$$T_d \approx -\frac{\varphi(\omega_1 T_s)}{\omega_1 T_s} = -\frac{\arg G_{ap}(e^{i\omega_1 T_s})}{\omega_1 T_s} \tag{4.22}$$

for a small ω_1 and adding a 1 because of the extra time-delay that is created by the sampling:

$$\hat{T}_d = -\frac{\arg G_{ap}(e^{i\omega_1 T_s})}{\omega_1 T_s} + 1; \quad \omega_1 \text{ small} \tag{4.23}$$

This approach is described in [Hor00, IHD01b]. (In [Hor00, IHD01b] a different derivation is done.) In [Hor00, IHD01b] $\omega_1 = 10^{-4}$ is suggested. The method in Equation 4.23 is in this thesis called DAP (Discrete-time Allpass Phase). Other forms of approximations of the derivative should also be possible.

4.2.6 Problems with the DAP method

In order to better understand how the DAP method works, we will in this section in detail calculate the DAP estimate l for two simulated trials. In one of the trials the method will work well but in the other it will fail completely.

By looking at how the phase of a model $G(e^{i\omega T_s})$ can be calculated from the poles and zeros of $G(z)$ we can get some insight into what happens in the calculation of the time-delay estimate in Equation (4.23).

If we have a rational discrete-time transfer function

$$G(z) = \beta \frac{(z - z_1) \dots (z - z_m)}{(z - p_1) \dots (z - p_n)}, \tag{4.24}$$

its frequency function will be

$$G(e^{i\omega T_s}) = \beta \frac{(e^{i\omega T_s} - z_1) \dots (e^{i\omega} - z_m)}{(e^{i\omega T_s} - p_1) \dots (e^{i\omega T_s} - p_n)}. \tag{4.25}$$

Let $\bar{Z}_k(\omega)$ be the vector $(e^{i\omega T_s} - z_k)$ from z_k to the point $e^{i\omega T_s}$ on the unit circle. Write $\bar{Z}_k(\omega)$ in polar form as $\bar{Z}_k(\omega) = Z_k(\omega)e^{i\psi_k(\omega)}$, where $Z_k(\omega)$ is the absolute value and $\psi_k(\omega)$ the phase of $\bar{Z}_k(\omega)$. We do the same for the poles. $\bar{P}_k(\omega)$ is the vector $(e^{i\omega T_s} - p_k)$ from p_k to the point $e^{i\omega T_s}$ on the unit circle and $\bar{P}_k(\omega) = P_k(\omega)e^{i\vartheta_k(\omega)}$. Then, we can write the phase of $G(e^{i\omega T_s})$ as

$$\arg G(e^{i\omega T_s}) = \arg\beta + \sum_{k=1}^{m}\psi_k(\omega) - \sum_{k=1}^{n}\vartheta_k(\omega). \tag{4.26}$$

Now, if we write the phase of the allpass part $G_{\text{ap}}(z)$ of an identified model as in Equation (4.26) and use Equation (4.23) we arrive at the expression

$$\begin{aligned}\hat{T}_d &= 1 - \frac{\text{unwrap}(\arg G_{\text{ap}}(e^{i\omega_1 T_s}))}{\omega_1 T_s} = \\ &= 1 - \frac{1}{\omega_1 T_s}\text{unwrap}\left(\arg\beta + \sum_{k=1}^{m}\psi_k(\omega_1 T) - \sum_{k=1}^{n}\vartheta_k(\omega_1 T_s))\right)\end{aligned} \tag{4.27}$$

for the time-delay estimate. unwrap$(\cdot)$ means replacing a phase outside $[-\pi,\pi]$ with the corresponding phase inside $[-\pi,\pi]$.

We will now give an example of when the DAP method works well. The linear continuous-time system

$$G_2(s) = e^{-9s}\frac{1}{(s+1)(0.1s+1)} \tag{4.28}$$

(same as system G_2, Equation (7.3) in Section 7.3) was simulated by the function `lsim` in [CST] in continuous-time with a random binary input signal with frequency contents between 10%-30% of the Nyquist frequency. The signal was generated by the function `idinput` in [SIT] and was the same as the signal "RBS 10-30% " in Section 7.3. To the output signal, white Gaussian noise was added and the signal-to-noise ratio (SNR) was 10. The sampling interval was $T_s = 1$ and $\omega_1 = 10^{-4}$. A Laguerre model (Section 4.2.3) was identified from the input-output data and the DAP method was used to estimate the time-delay. We will now step for step go through the calculation of $\hat{T}_d$.

The simulation resulted in the poles and zeros of the identified Laguerre model depicted in Figure 4.8. Table 4.1 lists $\psi_k(\omega_1)$ and $\vartheta_k(\omega_1)$.

The time-delay estimate is

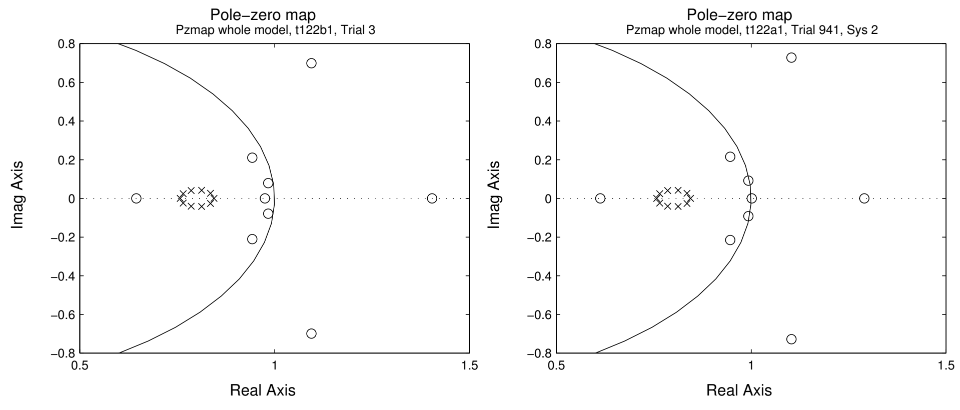
$$\begin{aligned}\hat{T}_d &= 1 - \frac{\text{unwrap}(\arg G_{\text{ap}}(e^{i\omega_1 T_s}))}{\omega_1 T_s} = \\ &= 1 - \frac{1}{\omega_1 T_s}\text{unwrap}\left(\arg\beta + \sum_{k=1}^{m}\psi_k(\omega_1 T) - \sum_{k=1}^{n}\vartheta_k(\omega_1 T_s))\right) = \\ &= 1 - 10^4\cdot\ \text{unwrap}\,(3.14159 - 1.70495 + 1.70491 + 3.14135 \\ &\quad -0.86837 + 0.86813 - 0.00035) = 1 - 10^4\cdot\ \text{unwrap}\,(6.28231) \\ &= 1 - 10^4\cdot(-0.00087114) = 9.7114,\end{aligned} \tag{4.29}$$

which is a good estimate since the true time-delay is 10.

Table 4.1 $\psi_k(\omega_1)$ and $\vartheta_k(\omega_1)$ in Equation (4.26) for $\omega_1 = 10^{-4}$ for the successful estimation (left,$\arg\beta = 3.14159$), the failed estimation (middle, $\arg\beta = 0$, not the same estimated model as for the successful estimate) and after removing the real zero just outside the unit circle (right, $\arg\beta = 3.14159$, the same estimated model as for the failed estimate).

Successful estimation		Failed estimation		After zero guarding	
$\psi_k(\omega_1)$	$\vartheta_k(\omega_1)$	$\psi_k(\omega_1)$	$\vartheta_k(\omega_1)$	$\psi_k(\omega_1)$	$\vartheta_k(\omega_1)$
-1.70495	0.86837	-1.71186	0.84682	-1.71186	0.84682
1.70491	-0.86813	1.71182	-0.84658	1.71182	-0.84658
3.14135	0.000347	3.14125	0.000445	3.14125	0.000445
		3.07613	*0.06556*		

Figure 4.8 Pole-zero plots of two identified Laguerre models. System G_2 and input signal RBS 10-30% (Section 7.3). SNR =10. No prewhitening (Section 7.3) or zero guarding (Section 4.2.7). The poles of the models should all be located at $\alpha = 0.8$ but due to well-known numerical problems with multiple poles they are somewhat spread. Left plot: Successful time-delay estimation ($\hat{T}_d = 9.7114$). Right plot: Failing time-delay estimation ($\hat{T}_d = 1321.86$). This simulation was one of only 3 out of 1024 simulations with failing time-delay estimation for SNR = 10. With a lower SNR the percentage of failing estimations is much higher. See Section 4.2.7. (t122a1)



a. Successful ***b.*** Failing

We will now study a case when the time-delay estimation fails. The only difference to the successful trial in the simulation setup is a different noise realization. The zeros and poles of the identified Laguerre model are shown in Figure 4.8. We note that the zero on the real axis just inside the unit circle in the successful simulation has moved to just outside the unit circle. We will later see what impact this will have on the time-delay estimate. Table 4.1 lists $\psi_k(\omega_1)$ and $\vartheta_k(\omega_1)$.

The time-delay estimate will now be

$$\begin{aligned}
\hat{T}_d &= 1 - \frac{\text{unwrap}(\arg G_{\text{ap}}(e^{i\omega_1 T_s}))}{\omega_1 T_s} = \\
&= 1 - \frac{1}{\omega_1 T_s}\text{unwrap}\left(\arg\beta + \sum_{k=1}^{m}\psi_k(\omega_1 T) - \sum_{k=1}^{n}\vartheta_k(\omega_1 T_s))\right) = \\
&= 1 - 10^4 \cdot \text{ unwrap}\,(0 - 1.71186 + 1.71182 + 3.14125 + 3.07613 \\
&\quad -0.84682 + 0.8465 - 0.000445 - 0.06556) = \\
&\quad 1 - 10^4 \cdot \text{ unwrap}\,(6.1511) = 1 - 10^4 \cdot (-0.132086) = 1321.9. \qquad (4.30)
\end{aligned}$$

As can be seen, the estimate is completely wrong.

4.2.7 A solution to the problem with the DAP method

As already mentioned in the previous section, a difference between the identified models in the successful and the failing trials was that the zero on the real axis just inside the unit circle in the successful trial had moved to outside the unit circle. Let us see what happens if we simply remove the moved zero from the model with the failing estimate. This results in the $\psi_k(\omega_1)$ and $\vartheta_k(\omega_1)$ in Table 4.1. The remaining $\psi_k(\omega_1)$ and $\vartheta_k(\omega_1)$ values in Table 4.1 are similar to the ones for the successful estimation in Table 4.1.

The time-delay estimate will now be

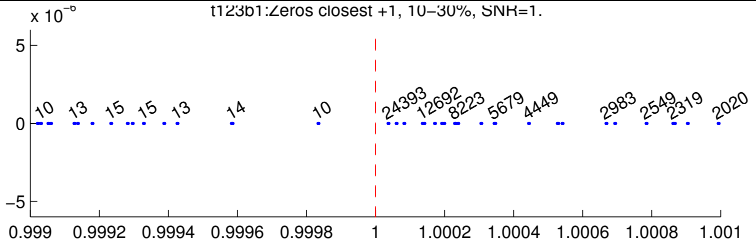
$$\begin{aligned}
\hat{T}_d &= 1 - \frac{\text{unwrap}(\arg G_{\text{ap}}(e^{i\omega_1 T_s}))}{\omega_1 T_s} = \\
&= 1 - \frac{1}{\omega_1 T_s}\text{unwrap}\left(\arg\beta + \sum_{k=1}^{m}\psi_k(\omega_1 T) - \sum_{k=1}^{n}\vartheta_k(\omega_1 T_s))\right) = \\
&= 1 - 10^4 \cdot \text{ unwrap}\,(3.14159 - 1.71186 + 1.71182 + 3.14125 \\
&\quad -0.84682 + 0.84658 - 0.000445) = 1 - 10^4 \cdot \text{ unwrap}\,(6.28212) \\
&= 1 - 10^4 \cdot (-0.00106586) = 11.6586, \qquad (4.31)
\end{aligned}$$

which is an acceptable estimate. It is much better than in Section 4.2.6 with the failing estimation. It seems like the only significant difference between the models with successful and failing estimation is if a certain zero is inside or outside the unit circle.

We will now study the uncertainty in the zeros and the time-delay estimate. We note in Equation (4.26) that if we move a zero from inside to outside of the unit circle when $\omega = 0$, then $\arg G(e^{i\omega T_s})$ will increase by π. If ω is not zero but very small, the phase increase will be between 0 and π. When this extra phase is divided by a very small ω_1 in Equation (4.23), the time-delay estimate will get a large bias that in most cases will dominate the estimate.

Figure 4.9 displays the time-delay estimate for different locations of the zero closest to +1. If the zero is inside the unit circle the estimates are reasonable. Outside the unit circle the estimates are poor. The worst estimates are close to the unit circle but outside it. Farther away from the unit circle (outside it) the estimate become better and better. The maximum possible estimation error occurs for the maximum phase error in $\arg G_{ap}(e^{i\omega_1 T_s})$, which is π. The estimation error will then be $\tilde{T}_d = \hat{T}_d - T_d = \pi/\omega_1 = 3.1416/10^{-4} = 31416$. In this calculation we have assumed that only one zero falls on the wrong side of the unit circle.

Figure 4.9 Time-delay estimate for different locations of the zero closest to +1 for a Laguerre model.(t123b1)



The reason for zeros falling on the incorrect side of the unit circle is the noise. Figure 4.10 shows an example of the spread of zeros and poles due to the noise.

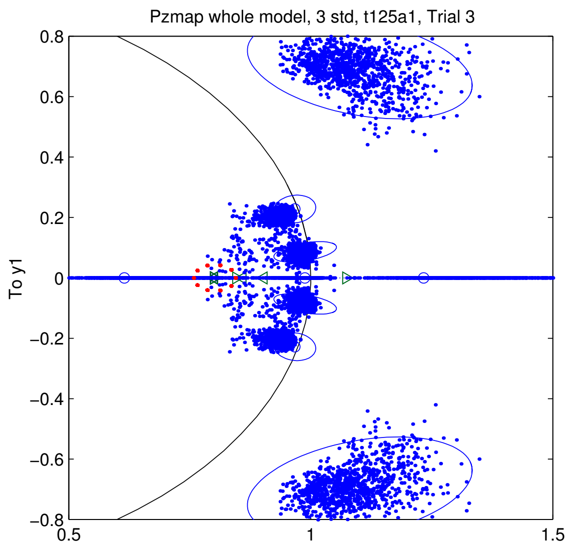
We will now show a solution to the problem with "false zeros" of the DAP method. It appears that moving zeros located close to but outside the unit circle (back) to the inside of the unit circle is a solution to the problem with the DAP method. The motivation is that we assume that these zeros actually should be located inside the unit circle. Since we only need the allpass part in the DAP methods, we just remove some zeros outside the unit circle without putting them somewhere inside the unit circle. We call this technique for *zero guarding*.

Now we have the following questions:

- Shall only the zeros outside the unit circle close to +1 or all zeros outside and close to the unit circle be (re)moved?
- How many zeros shall be (re)moved? Only one or perhaps several?
- What is meant by "close"?

In the Section 8.2.1 we will use a statistical technique to answer these questions.

Figure 4.10 Pole-zero plot with estimated uncertainty regions (3 standard deviations) and with zeros and poles from 1024 simulated trials for SNR = 1. As can be seen, the risk of a zero falling on the wrong side of the unit circle is significant. The simulation setup is as in Section 7.3 with system G_2 in open loop, signal RBS 10-30%, no prewhitening and no zero guarding (see Section 4.2.7). (t125a1)



4.3 Laguerre Domain Approximation Methods

In *Laguerre domain approximation methods* an approximation of the system including the time-delay is represented in the Laguerre domain. This means that the input, output and noise signals are transformed into the Laguerre domain, i.e. in the basis where the basis functions are Laguerre functions. From this system representation it is not as easy as in the time and frequency domain to get the time-delay but can be done as we will see in this section. The signals and the system can be represented either in continuous-time or discrete-time. We describe only the discrete-time case. The continuous-time case is described in [Fis99, FM99c, FM99a].

4.3.1 The discrete-time Laguerre domain

By taking the inverse Z-transform ($\mathcal{Z}^{-1}\{\cdot\}$) of the discrete-time Laguerre function in the frequency domain (Equation 4.14) we get the time domain discrete-time Laguerre functions

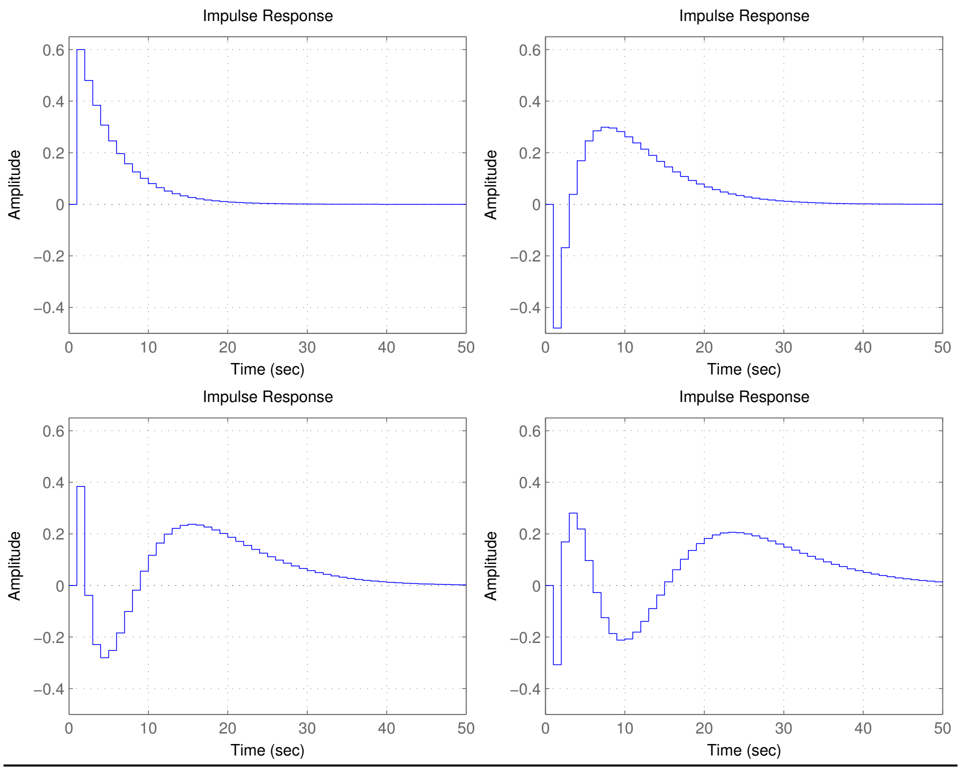
$$l_k = \mathcal{Z}^{-1}\{L_k(z)\} \tag{4.32}$$

They are orthonormal in $l_2(0,\infty)$. The space $l_2[a,b]$ is the space which consists of all square summable discrete-time functions defined on an interval $[a,b]$ with the inner product defined as

$$\langle f, g\rangle = \sum_{t=0}^{\infty} f(t)g(t) \tag{4.33}$$

for $f, g \in l_2[a, b]$. The first four discrete-time Laguerre functions are depicted in figure 4.11.

Figure 4.11 MATLAB plot of the first four Laguerre functions in discrete time. $k = \begin{matrix}[1] & [2] \\ [3] & [4]\end{matrix}$, $\alpha = 0.8$, sampling interval $T_s = 1$. (l1disc-)



A discrete-time signal $w(t)$ can now be represented by

$$w(t) = \sum_{k=0}^{\infty} w_k l_k(t), \tag{4.34}$$

where the Laguerre coefficients w_k are given by

$$w_k = \langle w(t), l_k(t)\rangle\,. \tag{4.35}$$

This means that the coefficients w_k describe (linearly) independent properties of the signal $w(t)$. Since all real signals have finite extent in the time, they belong to l_2 (for discrete-time) and can therefore be represented exactly by an infinite sum as in Equations (4.34)-(4.35).

4.3.2 System representation in the discrete-time Laguerre domain

For the discrete-time signal model

$$y(t) = u(t-d) + n(t),$$

where t and d only can be integer valued [Fis99], the relation between the input, output and noise signals in the discrete-time Laguerre domain is [Fis99, FM99c]

$$y_k = \sum_{l=0}^{k} \phi_{k,l} u_l + n_k, \tag{4.36}$$

where y_k, u_l and n_k are the Laguerre coefficients for the output, input and noise signals, respectively. Thus, the relation between the input and the output in the Laguerre domain can be seen as a time-varying linear filter. The time-varying impulse response $\phi_{k,l}$ is given by [Fis99, FM99c]

$$\phi_{k,l} = \langle GL_l\,,\, L_k \rangle\,, \tag{4.37}$$

where $G = G(z) = z^{-d}$ and $\langle \cdot, \cdot \rangle$ is the scalar product in the Z domain [FM99b].

The relation in Equation (4.36) can be rewritten as a linear regression [Fis99, FM99c, FM99b]

$$y_k = \varphi_k^T \Theta + n_k, \tag{4.38}$$

with the regression vector $\varphi_k = [\varphi_{k,1}, \ldots, \varphi_{k,N+1}]^T$, whose elements are

$$\begin{array}{lcll}
\varphi_{k,1} & = & u_0 & \\
\varphi_{k,l+1} & = & \frac{(1-\alpha^2)^l}{l!(l-1)!} \sum_{m=0}^{k-l} (-1)^{k+l-m} \alpha^{k-2l-m} \frac{(k-m-1)!}{(k-m-l)!} u_m, & k \geq l > 0 \\
\varphi_{k,l+1} & = & 0 & N \geq l > k
\end{array} \tag{4.39}$$

$N+1$ is the total number of Laguerre coefficients and α is the Laguerre pole (Equation 4.14). The parameter vector Θ is [FM99b]

$$\Theta = \left[\begin{array}{cccc} 1, & d, & \ldots, & d(d-1)\cdots(d-(N-1)) \end{array}\right]^T \alpha^d. \tag{4.40}$$

Here it is seen where the time-delay T_d comes in.

In [Fis99, FM99c] a method to estimate the time-delay is suggested: Let

$$Y = [y_0, \ldots, y_N]^T \tag{4.41}$$

and

$$\Phi = [\varphi_0, \dots, \varphi_N]^T. \tag{4.42}$$

Since Φ is a square nonsingular matrix, Θ could be estimated by

$$\hat{\Theta} = \Phi^{-1} Y. \tag{4.43}$$

$$\hat{d} = \left(\mathbf{1}^T \mathbf{1}\right)^{-1} \mathbf{1}^T Y_d = \frac{1}{N} \mathbf{1}^T \hat{D} + \frac{N-1}{2}, \tag{4.44}$$

where $\mathbf{1} = [1, \dots, 1]^T$, $Y_d = \hat{D} + [0, 1, \dots, N-1]^T$ and $\hat{D} = \operatorname{diag}\left(\underline{\hat{\Theta}}\right)^{\dagger} \overline{\hat{\Theta}}$. The vector $\underline{\hat{\Theta}}$ is given by the vector $\hat{\Theta}$ without the last element and $\overline{\hat{\Theta}}$ by $\hat{\Theta}$ without the first element. The symbol $\dagger$ means pseudoinverse. In [FM99b] also an other method to estimate the time-delay is suggested. See also the next section.

4.3.3 Two discrete-time Laguerre domain time-delay estimation algorithms

In [Fis99, FM99c, FM99b] two algorithms for the implementation of time-delay estimation based on the system representation (4.38), (4.39)-(4.40) in the discrete-time Laguerre domain are given. One of them uses Equation (4.44). They also have some features to enhance the numerical properties.

The matrix Φ will not be well-conditioned for large N [Fis99]. The numerical properties of Φ are improved by using SVD (singular value decomposition [HJ91]) to extract the most significant part of Φ. See [Fis99, FM99a, FM99b] for details. The number of singular values (the largest ones) to retain is a user-defined parameter. It is in this thesis set to 5, which is chosen after some simple tests.

The time-delay d can be estimated in two ways, here called tausvd and taulp1. In tausvd, d is estimated by least squares as in Equation (4.44). In taulp1 the Θ estimate (Equation (4.43)) is further improved by an iterative algorithm called MIRLS (Modified Iteratively Reweighted Least Squares) [FM99b, CCS94] before estimating the time-delay by Equation (4.44). See Algorithms 3-4.

Several parameters must be selected by the user, most importantly the Laguerre pole α and the number $N+1$ of Laguerre functions to use. These must be selected to suit the input and output signals and the available execution time [Fis99, FM99c, FM99b, Bjö03e].

It is also possible to use approximation methods in other domains, e.g. the Kautz domain [Kau54, Wah94, SBS97].

4.3.4 Approximating the signals with the Laguerre transform

A necessary condition for the Laguerre domain approximation estimation methods to be successful is that the input and output signals can be represented accurately

Algorithm 3 tausvd: An implementation of Laguerre domain approximation method.

1. Optimize the pole α given the input output signals: "minimizing the squared equation error between the input signal and its approximation by a truncated ($N = 16$) Laguerre series with respect to α" [Fis99, FM99b, FM99c].

2. Compute the regression matrix Φ (4.39), (4.42) and the Laguerre spectrum Y of the output signal from the input and output signals (Eq. (4.35)).

3. Retain the most significant part Φ_s of the regression matrix Φ by SVD: $\Phi = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & \\ & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}$ and $\Phi_s = U_1 \Sigma_1 V_1^T$.

   The diagonal matrix Σ_1 has the largest singular values on the diagonal. The number of used singular values in Σ_1 is a user-selected parameter.

4. Estimate the parameter vector: $\hat{\Theta} = \Theta_s^\dagger Y_s$, where $Y_s = [Y^T(1:s), 0, \ldots, 0]^T$.

5. Estimate the time-delay : $\hat{d} = \left(\mathbf{1}^T \mathbf{1}\right)^\dagger \mathbf{1}^T Y_d$

Algorithm 4 taulp1: Another implementation of Laguerre domain approximation method.

1. Estimate the parameter vector Θ as in step 1-4 in algorithm 3.

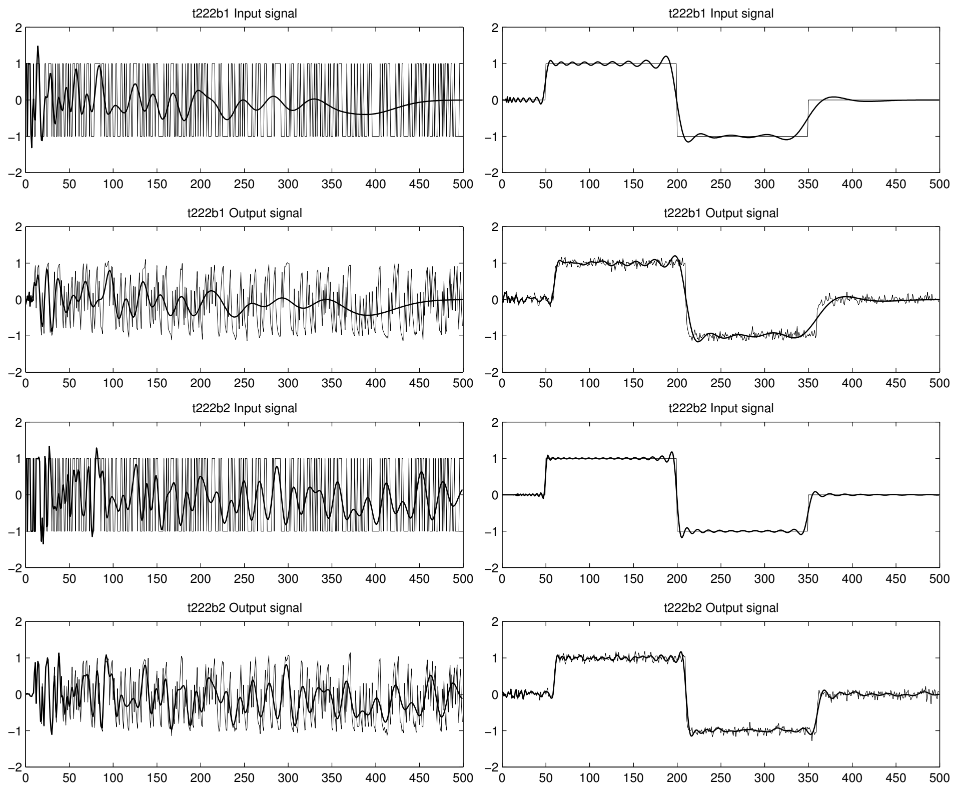
2. Use the estimated parameter vector $\hat{\Theta}$ as start value to the MIRLS [FM99b, CCS94] to improve the estimate of the parameter vector.

3. Estimate the time delay as in step 5 in algorithm 3.

in the Laguerre domain. Figure 4.12 displays the original and the Laguerre approximated input and output signals for two input signal types. The Laguerre approximation behaves similarly to a low-pass filtering but it works best at the beginning of the signals. We see that:

- The signal Steps is easy to approximate. The RBS signals are very hard.
- The approximations become better with more Laguerre coefficients.

Figure 4.12 Original and Laguerre approximated input and output signal for RBS 0-100% (left) and Steps (right) with $N+1 = 51$ (above) and $N+1 = 150$ (below). Laguerre pole $\alpha = 0.8$, G_2, SNR=100. The signals and the system are defined in Section 7.3. $N+1$ is the number of used Laguerre functions. (t222b1-t222b2)



5 Explicit Time-Delay Parameter Methods

In *explicit time-delay parameter methods*, the time-delay is an explicit parameter to be estimated in the model. In Section 5.1 the time-delay is a continuous parameter in a continuous-time model, while in Section 5.2 a discrete parameter in a discrete-time model. Section 5.3 describes some sampling methods.

5.1 Continuous-Time One-Step Explicit Methods

In *continuous-time one-step explicit methods*, the time-delay is modeled and estimated as a continuous parameter in a continuous-time model. The time-delay is thus not restricted to be a multiple of the sampling interval. See for example [NL91, Lju02].

5.1.1 The time-delay estimation methods

Here we will describe a group of methods consists of estimating the time-delay as a continuous parameter with the prediction error method (Section 4.1.5 and[Lju99]) in some simple model structures [Lju02] which are often used in process industry. We sometimes call this type of methods *idproc methods.* The process models are:

idproc1. A first order system with time-delay.

idproc2. A second order system with real poles and time-delay.

idproc3. A second order system with complex poles and time-delay.

idproc4. A third order system with real poles and time-delay.

idproc5. A third order system with two complex poles, one real pole and time-delay.

In the prediction error method (PEM) [Lju99]:

$$\hat{\theta}_N = \arg\min_{\theta} V_N(\theta) = \arg\min_{\theta} \frac{1}{N}\sum_{t=1}^{N} \frac{1}{2}\varepsilon^2(t,\theta), \tag{5.1}$$

where $\varepsilon(t|\theta) = y(t) - \hat{y}(t|\theta)$ is the prediction error of the model. See also Section 4.1.5.

Since the loss function $V_N(\theta)$ in Equation (5.1) has several minima [FMS96, NL91], the initialization of the optimization problem (5.1) must be done carefully [Lju02]. See [Lju03, BL03] for the implementation used in the simulations of this thesis. An alternative to accurately initialize the time-delay could be to perform a low-pass filtering. See Section 5.1.3.

The article [NL91] contains formulas for zoh sampling of a system with time-delay in state space form including a noise model. It also contains MATLAB code for how to implement a PEM algorithm.

5.1.2 Local minima of a first order system

We will here study the phenomenon that the loss function $V_N(\theta)$ in Equation (5.1) has several minima more closely.

Assume, both the true system and the model is of the same structure: a first order system with time-delay:

$$\bar{G}(s) = \frac{K}{s+\beta}e^{-sL}. \tag{5.2}$$

The system sampled with the sampling interval T_s using zero-order-hold (zoh) sampling [ÅW84] becomes

$$y(k) = \frac{b_1 q^{-1} + b_2 q^{-2}}{1 + a_1 q^{-1}} u(k) = G(q)u(k) \tag{5.3}$$

with

$$b_1 = K\frac{e^{-\beta(T_s-L)}}{\beta}(e^{-\beta(T_s-L)} - 1) \tag{5.4}$$

$$b_2 = K\frac{e^{-\beta(T_s-L)}}{\beta}(1 - e^{-\beta L}) \tag{5.5}$$

$$a_1 = -e^{-\beta T_s}. \tag{5.6}$$

We will denote the true system with $\bar{G}_o(s)$ (in continuous-time) and $G_o(q)$ (the sampled discrete-time version of $\bar{G}_o(s)$) and its parameters with b_{10}, b_{20} and a_{10}.

The model will be denoted by $\bar{G}(s|L)$ and $G(q|L)$ and its parameters with $b_1(L)$, $b_2(L)$ and a_1. If no other parameters than the time-delay differs between the true system and the model, then $a_{10} = a_1$(Equation (5.6) does not depend on L) . Both the true output signal and the simulated output signal of the model are generated according to

$$y(k) = G(q)u(k) + H(q)e(k).$$

We assume that the noise system is $H(q) = 1$. This gives an output error structure. The noise $e(k)$ is zero mean and white with variance λ_e. The input signal $u(k)$ is zero mean white noise with variance λ_u.

As the loss function to minimize we choose

$$V(L) = \mathrm{E}\{\varepsilon^2(k|L)\} = \mathrm{Var}\{\varepsilon(k|L)\} = R_{\varepsilon\varepsilon}(0|L) \tag{5.7}$$

where the residuals $\varepsilon(k|L)$ are given by

$$\varepsilon(k|L) = y(k) - \hat{y}(k|L) = (G_o(q) - G(q|L))u(k) + e(k)$$

with $\hat{y}(k|L) = G(q|L)u(k)$ being the one-step-ahead predictor of $y(k)$ (OE structure) given L [Lju99].

In normal circumstances, $\frac{1}{2}V(L)$ is what the loss function $V_N(\theta)$ (Equation (5.1)) in the prediction error method (PEM) converges to when the number of data $N \to \infty$ [Lju99, Th.8.2, p.254]. See also Section 4.1.5.

Assume now that the model and the true systems are stable and equal except for the time-delay, i.e $K = K_0$ and $\beta = \beta_0 > 0$. Assume that the true time-delay L_o fulfills $0 < L_o < T_s$. Then the loss function for $0 < L < T_s$ can be calculated as

$$V(L) = \frac{\tilde{b_1}^2(L) - 2a_1\tilde{b_1}(L)\tilde{b_2}(L) + \tilde{b_2}^2(L)}{1 - a_1^2}\lambda_u + \frac{1}{1-a_1^2}\lambda_e \tag{5.8}$$

with $\tilde{b_1}(L) = b_{10} - b_1(L)$ and $\tilde{b_2}(L) = b_{20} - b_2(L)$.

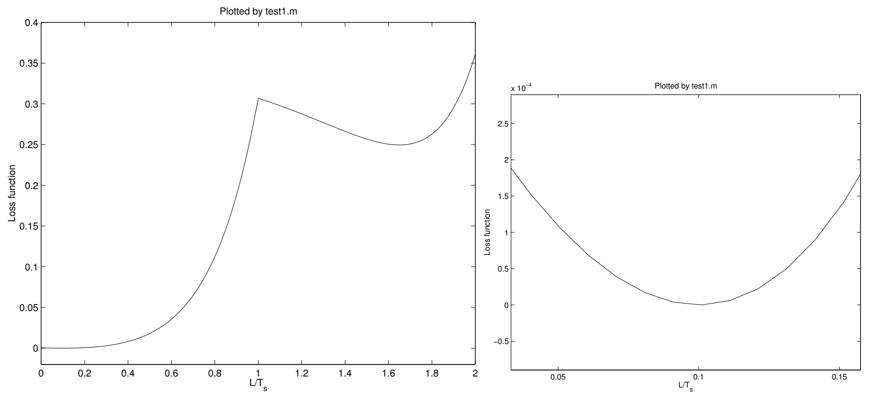
By inserting Equations (5.4)-(5.6) into Equation (5.8) we get the following expression

$$\begin{aligned} V(L) &= \left(e^{-\beta(-L_0+L)} - 1\right)\left(e^{-\beta(-L_0+L)} - 1 - e^{-\beta(T_s-L_0+L)} + e^{-\beta T_s}\right) \\ &\quad \cdot \frac{2K^2 e^{2\beta(-T_s+L)}}{\beta^2\left(1+e^{-\beta T_s}\right)\left(1-e^{-\beta T_s}\right)} \end{aligned} \tag{5.9}$$

The loss function for $T_s < L < 2T_s$ is

$$\begin{aligned} V(L) &= \frac{-2b_2(L-1)a_1^2 b_{10} + 2b_2(L-1)a_1 b_{20} + 2b_1(L-1)a_1 b_{10}}{1-a_1^2}\lambda_u \\ &+ \frac{-2b_{20}a_1 b_{10} - 2b_1(L-1)b_{20} + b_1(L-1)^2}{1-a_1^2} \\ &+ \frac{-2b_2(L-1)a_1 b_1(L-1) + b_{10}^2 + b_{20}^2 + b_2(L-1)^2}{1-a_1^2}\lambda_u + \frac{1}{1-a_1^2}\lambda_e \end{aligned}$$

Figure 5.1 The loss function (without the term due to the noise $e(k)$) for $0 < L < 2T_s$ when the true time-delay is $0.1T_s$. The right graph is a zoom-in on the left one. ($K = K_0 = 1$, $\beta = \beta_0 = 2$) (idproclossfunc4-6)



Figures 5.1-5.3 show the loss function $V(L)$ for $0 < L < 2T_s$ when the true time-delay is $0.1T_s$, $(2/3)T_s$ $0.9T_s$ and $0.9999T_s$. We see that there is more than one minimum but within $0 < L < T_s$ it seems to be only one minima. This means that we have to know the integer part of L/T_s. This is in accordance with [NL91].

In Figure 5.3 we also see that for a true delay very close to a multiple of the sampling interval ($0.9999T_s$ in the figure) we must start very close to the true value, if starting on an unsuitable side, not to miss it.

We will now prove that there is exactly one minimum of $V(L)$ within $0 < L < T_s$. The first derivative of the loss function in Equation (5.9) is

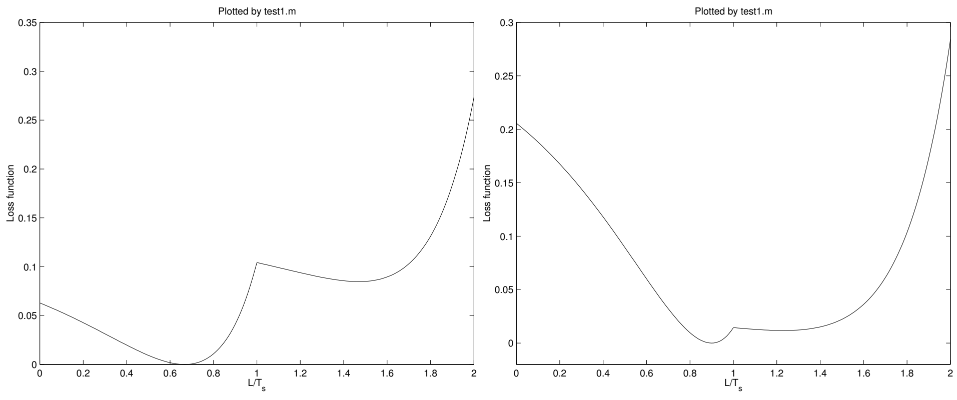
$$\begin{aligned}\frac{dV(L)}{dL} &= -4\,\frac{K^2\left(e^{\beta(L_0+L-2T_s)} - e^{\beta(L_0+L-3T_s)} - e^{2\beta(-T_s+L)} + e^{\beta(-3T_s+2L)}\right)}{\beta\left(1+e^{-\beta T_s}\right)\left(1-e^{-\beta T_s}\right)} \\ &= -4\,\frac{K^2 e^{\beta(L-2T_s)}\left(\left(e^{\beta L_0} - e^{\beta L}\right) - \left(e^{\beta(L_0-T_s)} - e^{\beta(L-T_s)}\right)\right)}{\beta\left(1+e^{-\beta T_s}\right)\left(1-e^{-\beta T_s}\right)} \\ &= -4\,\frac{K^2 e^{\beta(L-2T_s)}\left(\chi(L_0,L) - \psi(L_0,L)\right)}{\beta\left(1+e^{-\beta T_s}\right)\left(1-e^{-\beta T_s}\right)}, \qquad (5.10)\end{aligned}$$

where $\chi(L_0, L) = (e^{\beta L_0} - e^{\beta L})$ and $\psi(L_0, L) = (e^{\beta(L_0-T_s)} - e^{\beta(L-T_s)})$. We notice that

$$\begin{aligned}&\chi(L_0,L) > 0 &&\text{when } L < L_0\\ &\chi(L_0,L) < 0 &&\text{when } L_0 < L\\ &\psi(L_0,L) > 0 &&\text{when } L < L_0\\ &\psi(L_0,L) < 0 &&\text{when } L_0 < L\end{aligned}$$

and that $|\chi(L_0, L)| > |\psi(L_0, L)|$ since $T_s > 0$ and the exponential function has a monotonically increasing derivative. From this follows $\chi - \psi > 0$ when $L < L_0$

Figure 5.2 The loss function (without the term due to the noise $e(k)$) for $0 < L < 2T_s$ when the true time-delay is $(2/3)T_s$ (left) and $0.9T_s$ (right). ($K = K_0 = 1$, $\beta = \beta_0 = 2$) (idproclossfunc1-3)



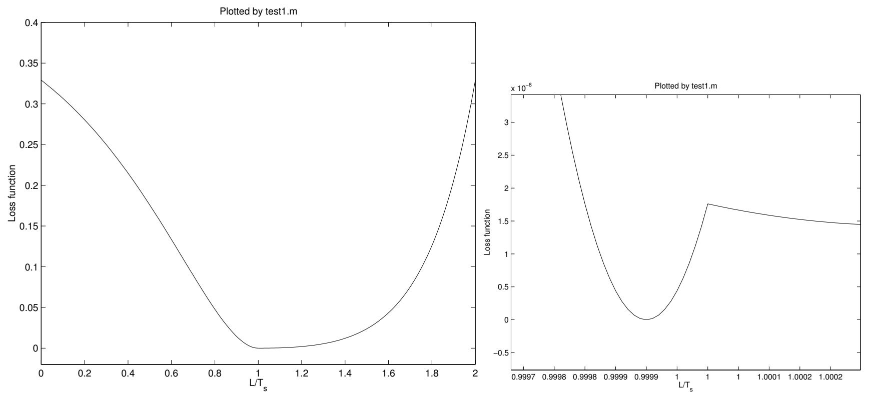
and $\chi - \psi < 0$ when $L_0 < L$. This together with the fact, that all other factors in Equation (5.10) are positive except for the constant -4, means that $\frac{dV(L)}{dL} < 0$ for $L < L_0$ and $\frac{dV(L)}{dL} > 0$ for $L > L_0$. The loss function $V(L)$ is thus quasi-convex [BV03] and has only a single minimum within $0 < L < T_s$, which is at L_0. This is valid for the assumptions above., e.g. the input signal is white noise. The quasi-convexity within the sampling interval means that no matter how close to the true time-delay the optimization is started there are still cases when we will end in an incorrect local minimum. Therefore, not even close to the true time-delay the Cramer Rao lower bound [Kay93, JD93] is always useful.

For convex loss functions there exist very efficient optimization algorithms [BV03]. Therefore, we will now investigate if $V(L)$ is convex. The second derivative of the loss function in Equation (5.9) is

$$\begin{aligned}\frac{d^2V(L)}{dL^2} &= 4\,\frac{K^2\left(-e^{\beta(L_0+L-2T_s)} + e^{\beta(L_0+L-3T_s)} + 2\,e^{2\beta(-T_s+L)} - 2\,e^{\beta(-3T_s+2L)}\right)}{(1+e^{-\beta\,T_s})\,(1-e^{-\beta\,T_s})} \\ &= 4\,\frac{K^2 e^{\beta(L-2T)} g(L)}{(1+e^{-\beta\,T_s})\,(1-e^{-\beta\,T_s})}. \qquad (5.11)\end{aligned}$$

The loss function $V(L)$ is convex if and only if its second derivative is positive everywhere ($0 < L < T_s$). It is easy to see that all parts, except maybe the function $g(L)$, of the last formula in Equation 5.11 always are positive. Therefore we can concentrate on the function $g(L)$ when determining whether the second derivative is positive everywhere. In the example plot of $g(L)$ in Figure 5.4 we see that the loss function is not convex between the sampling instants. This can also nearly be seen in Figure 5.2 which is a plot of the corresponding loss function. In Figure 5.3, which depicts another loss function $V(L)$ it is easier to see that $V(L)$ is not convex.

Figure 5.3 The loss function (without the term due to the noise $e(k)$) for $0 < L < 2T_s$ when the true time-delay is $0.9999T_s$. The right graph is a zoom-in on the left one. $(K = K_0 = 1,\ \beta = \beta_0 = 2\ )$ (idproclossfunc5-2)



5.1.3 Widening the region of convergence by low-pass filtering

One idea to increase the region around the global minimum of $V(L)$ with no local minima would be to use a longer sampling period. This can be done by down-sampling the already collected signals. An essential part of this down-sampling would be, of course, a low-pass (LP) filtering to avoid sampling aliasing. For the sake of the region around the global minimum of $V(L)$ it should be possible to omit the actual reduction of the number of samples and only apply the LP filtering as in [FMS91]. Note, however, that down-sampling the input and output signals violates the prerequisites for the zoh sampling, which requires the input signal to be constant between the sampling instants. If we sample down after the data has been collected, the input signal was not constant between the new down-sampled sampling instants.

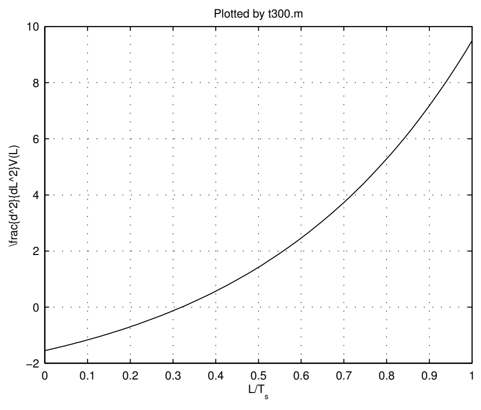
The article [FMS96] treats the question of multiple minima of the loss function and LP-filtering in more detail. The conclusions are that suitable LP-filtering increases the region of convergence but also reduces the convergence speed and increases the possible bias due to modeling errors of the rest of the system.

In [FMS96] the following sampled (zoh) system is used:

$$y(k) = G(q)q^{-d}u(k) + H(q)e(k), \tag{5.12}$$

where $G(q)$ and $H(q)$ are proper rational transfer functions. The system $G(q)$ is the sampled version of $\bar{G}_1(s) \cdot e^{-s\epsilon T_s}$, where $\bar{G}_1(s)$ is a proper rational function of s. The total time-delay is denoted $T_d = (d + \epsilon)T_s$ with d being an integer and $\epsilon \geq 0$. The noise $e(k)$ is a sequence of random and independent variables which are

Figure 5.4 The function $g(L)$, which is a part of the second derivative in Equation 5.11 of the loss function in Equation 5.9, for $0 < L < T_s$ when the true time-delay is $(2/3)T_s$. $(K = K_0 = 1,\ \beta = \beta_0 = 2\ )$ (t300)



uncorrelated with $u(k)$. The criterion to minimize is

$$V_2(d,\epsilon) = \sum_{k=-\infty}^{\infty} \varepsilon^2(k|d,\epsilon), \tag{5.13}$$

where $\varepsilon(k|d,\epsilon) = y(k) - \hat{y}(k|d,\epsilon)$ is the prediction error and $\hat{y}(k|d,\epsilon)$ is the one-step predictor of the system output. If $\varepsilon(k|d,\epsilon)$ is ergodic and has time extent T_ϵ, the functions $V(L)$ and $\frac{1}{T_\epsilon}V_2(d,\epsilon)$ in Equations (5.7) and (5.13), respectively, are the same.

If the region where the true time-delay can be located, the *uncertainty region*, is known, the following theorem [FMS96] can be used to check if the LP filter gives the required region without local minima.

Theorem 5.1
Assume the model structure (5.12), *the loss function* (5.13), *that the output signal* $y(k)$ *is sampled without aliasing, that the the input and output are filtered through the filter* $L(q)$ *before the time-delay estimation and that noise system* $H(q)$ *is known. The loss function is then*

$$V_2(\tilde{T}_d) = 2\left(R_{ww}(0) - \tilde{R}_{ww}(\tilde{T}_d/T_s)\right), \tag{5.14}$$

where $\tilde{T}_d = T_d - T_{d0}$ *is the error in the time-delay and* $R_{ww}(k)$ *is the autocorrelation function of the signal* $w(k) = L(q)H^{-1}(q)y(k)$. *The quantity* T_{d0} *is the true time-delay.* $\tilde{R}_{ww}$ *is given by*

$$\tilde{R}_{ww}(\tau) = \sum_{k=-\infty}^{\infty} R_{ww}(k)\frac{\sin(\tau\pi - k\pi)}{\tau\pi - k\pi}. \tag{5.15}$$

Note that in Equation (5.14), only the difference $\tilde{T}_d$ between the estimated and true time-delay matters and that $\tilde{T}_d$ is not restricted to multiples of the sampling interval but can take any value.

In the special case when $L(e^{i\omega T_s})H^{-1}(e^{i\omega T_s})\bar{G}(i\omega)$ is an ideal LP filter with cutoff frequency ω_b and the input signal is white noise, [FMS96] shows that the cutoff frequency of the LP filter should be selected as

$$\omega_b \lesssim \frac{4.49}{\Delta T_d},$$

where ΔT_d is the largest time-delay uncertainty. The rational part of the sampled system $G(q)$ is assumed known.

In the case of aliasing by the sampling and/or poor filtering, [FMS96] say that there may be many local minima, each belonging to a sampling interval.

Another way to accomplish the "widening of the sampling interval" without down-sampling could be to let the input signal be of low-pass type. This is similar but not equivalent to LP-filtering the input-output data (see [Lju99, p. 266-267, 466-468]).

5.2 Discrete-Time One-Step Explicit Methods

In *discrete-time one-step explicit methods,* the time-delay is modeled and estimated as a discrete parameter in a discrete-time model. Estimating several models, e.g. ARX models, with a complete set of time-delays and choosing the best is of this subclass ([Swa99, IHD00]). An exhaustive search is thus performed in the time-delay dimension. This subclass of methods is here also called *arxstruc type methods.* All methods described in this section are of this subclass. In Section 5.2.1 ARX models are utilized, in Section 5.2.2 OE models and in Section 5.2.3 prefiltered ARX models.

5.2.1 Arxstruc

In the method arxstruc several ARX models [Lju99]

$$A(q)y(t) = B(q)u(t - n_k) + e(t)$$

are estimated with PEM (Prediction Error Method) ([Lju99] and Section 4.1.5) for different time-delays n_k. PEM is here equal to Least Squares Estimation (LSE) [Lju99]. The delay whose model has the lowest loss function is chosen. The estimation is quick since the ARX model can be written as a linear regression and can be estimated by solving a linear equation system [Lju99]. See Algorithm 5 for MATLAB code using the Matlab System Identification Toolbox.

Algorithm 5 MATLAB code for arxstructd using the Matlab System Ident. Toolbox.

```
function dtEst = arxstructd(zIn);
zIn = [outSig, inSig];
na = 10; nb = 5; nkVec = 1:20;
nkMax = length(nkVec);
nn = [na*ones(nkMax,1), nb*ones(nkMax,1), nkVec'];
V = arxstruc(zIn,zIn,nn);
modelStruc = selstruc(V,0);
dtEst = modelStruc(3);
```

Algorithm 6 MATLAB code for oestructd using the Matlab System Ident. Toolbox.

```
function dtEst = oestructd(zIn);
zIn = [outSig, inSig];
nf = 2; nb = 1; nkVec = 1:20;
for nnn = 1:length(nkVec),
  model = oe(z,[nb nf nkVec(nnn)], 'Covariance','None')
  lossFunc(nnn) = model.NoiseVariance;
end%for
[minVal, nnnmin] =  min(lossFunc);
dtEst = nkVec(nnnmin);
```

5.2.2 Oestruc

The principle of the methods oestruc is the same as for arxstruc but OE (output error) models [Lju99]

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

are estimated instead of ARX models. To estimate OE models is much more computationally demanding than ARX models since a multidimensional optimization with a numerical search must be carried out [Lju99]. See Algorithm 6 for MATLAB code using the Matlab System Identification Toolbox.

5.2.3 Met1struc

The method oestruc gives a better result than arxstruc in the simulations presented in this thesis (see Sections 8.5.1 and 9.1-9.3). On the other hand, arxstruc has a much lower computation time than oestruc (Section 9.4). In this section we suggest a new time-delay estimation method with the aim to imitate oestruc but with much lower computational demands.

Assume the true system is given by

$$y = G_0 u + H_0 e\,,$$

where G_0 and H_0 are rational functions in the delay operator q^{-1}. The noise e is white. The model structure used to estimate the true system is

$$y = Gu + He\,,$$

where G and H also are rational functions.

The reason for the difference between OE and ARX model structures can be two-fold:

1. If the model structure G cannot exactly describe the true system G_0, the estimated model G will have a bias, even if the number of data $N \to \infty$. The OE and ARX model structures will behave differently in open loop [Lju99, Ex. 8.5, p. 268-269]:
   - The ARX model structure will give models with a good fit to the true system at high frequencies.
   - The OE model structure will give models with a good fit to the true system at low frequencies if the input is of low frequency character.
2. If the model structure G can exactly describe the true system G_0, again the OE and ARX model structures behave differently in open loop:
   - The ARX model structure will give models G with a bias if the noise model structure $1/A$ cannot describe the true noise system H_0. See [Lju99, eq. 8.63,8.69, p. 267].
   - The OE model structure will give estimates without bias. The bias of the OE model structure is thus not dependent on the noise model. See [Lju99, Eq. 8.71, p. 266].

We propose the time-delay estimation method in Algorithm 7. The motivation for this method is the following. Assume the true system is given by

$$y = G_0 u + H_0 e = \frac{B_0}{F_0} u + \frac{C_0}{D_0} e\,,$$

where G_0 and H_0 are rational functions and B_0, F_0, C_0 and D_0 are polynomials in the delay operator q^{-1}. The noise e is white.

Estimate a first model

$$y = \frac{B_1}{F_1} u + \frac{C_1}{D_1} e \tag{5.16}$$

where F_1, B_1, C_1 and D_1 are polynomials in q^{-1}, i.e. a Box-Jenkins model [Lju99].

Algorithm 7 Proposed time-delay estimation method (prefiltered arxstruc).

```
1. Estimate an ARMAX model $A_1(q)y(k) = B_1(q)u(k) + C_1(q)e(k)$.
2. Prefilter $u$ and $y$ through $1/C_1(q)$ .
3. Arxstruc gives an estimate of $n_k$.
```

The model C_1/D_1 will be an approximation of $H_0 = C_0/D_0$. Then, filter the output signal y through D_1/C_1 giving y_F:

$$y_F = \frac{D_1}{C_1}y = \frac{D_1}{C_1}\left(\frac{B_0}{F_0}u + H_0 e\right) = \frac{B_0}{F_0}\frac{D_1}{C_1}u + H_0\frac{D_1}{C_1}e = \frac{B_0}{F_0}u_F + H_0\frac{D_1}{C_1}e\,, \tag{5.17}$$

where $u_F = uD_1/C_1$ is the input signal u filtered through D_1/C_1. Now, if the polynomials F_1, B_1, C_1 and D_1 in the model structure 5.16 have high enough orders so that that they can exactly describe the true system and the input-output data is informative enough, then F_1, B_1, C_1 and D_1 will converge to the true values F_0, B_0, C_0 and D_0 as the number of data $N \to \infty$ [Lju99, p. 273]. When this happens, $D_1/C_1 = 1/H_0$ and Equation 5.17 simplifies to

$$F_0 y_F = B_0 u_F + e\,, \tag{5.18}$$

where e as before is white noise. This true system is obviously an ARX system. This means that if we estimate an ARX model $Ay_F = Bu_F + e$, there will be no bias due to an incorrect noise model. If an ARX model has high enough orders, we can get a model for the system G without bias (as in the case with an OE model of high enough orders).

Let us now assume that the true system instead has OE structure:

$$y = G_0 u + H_0 e = \frac{B_0}{F_0}u + e \Rightarrow F_0 y = B_0 u + F_0 e. \tag{5.19}$$

As this at the same time has ARMAX structure [Lju99] we estimate a first model

$$A_1 y = B_1 u + C_1 e\,, \tag{5.20}$$

where A_1, B_1 and C_1 are polynomials in q^{-1}. A_1 and C_1 will be approximations of F_0. If we filter the input and output signal through $1/C_1$ we will get the following true system

$$F_0 y_F = B_0 u_F + e\,, \tag{5.21}$$

which is also of ARX structure and can be approximated by an ARX model of high enough model order without bias.

Algorithm 8 met1struc, the prefiltered arxstruc.

```
1. Estimating a state space model by state space method.
2. Converting to $A_1(q)y(k) = B_1(q)u(k) + C_1(q)e(k)$ .
3. Prefiltering u and y through $1/C_1(q)$ .
4. Arxstruc gives an estimate of $n_k$.
```

Algorithm 9 Matlab code for met1struc using the Matlab System Identification Toolbox. The function arxstructd is described in Section 5.2.1.

```
function dtEst = met1structd(inSig, outSig, Ts)
order = 10;
modelSs = n4sid(iddata(outSig, inSig, Ts),...
  order,'cov','none');

[A,B,C,D,F] = polydata(idpoly(modelSs));
BFilt = 1;
AFilt = C;
uFilt = filter(BFilt,AFilt,inSig);
yFilt = filter(BFilt,AFilt,outSig);

zIn = [yFilt, uFilt];
na = 10; nb = 1; nkVec = 1:20;
dtEst = arxstructd(zIn,nkVec,na,nb);
```

Since the systems employed in most simulations in this thesis (Section 7.3) have OE structure (Equation (5.19)), we will filter the input and output signals through $1/C_1$ where C_1 is from Equation (5.20).

In Algorithm 7 the first step is to estimate an ARMAX model. Unfortunately, the standard way to estimate a model of this structure also requires a numerical search as with the OE model which we tried to avoid. Another way is to first estimate a state space model and then convert it to an ARMAX model. This conversion will be possible if the order of the state space model and the orders of the polynomials A_1, B_1 and C_1 are high enough to describe the true system. The state space model can be quickly estimated by a subspace method [Lju99]. See Algorithm 8 for the resulting method, which we call met1struc. Matlab code for met1struc is given in Algorithm 9. The order of the state space model is 10. This will hopefully be enough for most systems. The orders of A_1, B_1 and C_1 will depend on the order of the state space model.

5.3 Sampling Methods

Sampling methods utilize the sampling process to estimate the time-delay. For example, zero-order-hold (zoh) sampling of a system with subsample time-delays creates an extra zero [FMS91] as we will see soon.

5.3.1 Principles

We start with an example of sampling a simple system without time-delay. We consider the following continuous-time system:

$$\bar{G}(s) = \frac{K}{s+\beta}. \tag{5.22}$$

The system sampled with the sampling interval T_s using zero-order-hold (zoh) sampling, i.e. the input signal is constant between the sampling instants, (see[ÅW84, GL97]) becomes :

$$G(q) = \mu \frac{(1-p)}{(q-p)}, \tag{5.23}$$

with

$$\mu = K/\beta \tag{5.24}$$

and

$$p = e^{-\beta T_s}. \tag{5.25}$$

The letter q is the discrete-time-delay operator, i.e. q^{-1} is a delay of one sample.

When zoh sampling of a continuous-time system with a time-delay which is a fraction of a sampling interval is performed, a zero in the discrete-time system will arise due to the time-delay [ÅW84, p. 40]. We now will see what happens at the sampling when a time-delay is added to the previous example.

Assume that we have the same continuous-time system as in Equation (5.22) but now with a time-delay $0 < L < T_s$:

$$\bar{G}(s) = \frac{K}{s+\beta} e^{-sL} = \bar{G}_1(s) \cdot e^{-sL}. \tag{5.26}$$

The system sampled with the sampling interval T_s using zoh sampling becomes (cf. Section 5.1.2)

$$G(q) = \frac{b_1 q^{-1} + b_2 q^{-2}}{1 + a_1 q^{-1}} = \mu \frac{(1-p)}{(q-p)} \cdot \frac{((1-\alpha)q + \alpha)}{q}. \tag{5.27}$$

The quantities a_1, b_1 and b_2 are given by Equations (5.4)-(5.6) and μ and p by Equations (5.24) and (5.25). The quantity α is given by

$$\alpha = \frac{e^{\beta L} - 1}{1-p} p. \tag{5.28}$$

We see from Equations (5.23) and (5.27) that the result of the time-delay after the sampling is a pole in zero and an additional zero:

$$z_0 = -\frac{\alpha}{(1-\alpha)} \,. \tag{5.29}$$

If the time-delay is longer, $d \cdot T_s < dT_s + L < (d+1) \cdot T_s$, ($L$ is thus the fractional part of the time-delay), the sampled system will be [ÅW84, FMS91]

$$G(q) = \frac{b_1 q^{-1} + b_2 q^{-2}}{1 + a_1 q^{-1}} q^{-d} = \mu \frac{(1-p)}{(q-p)} \cdot \frac{((1-\alpha)q + \alpha)}{q} q^{-d}. \tag{5.30}$$

The only difference to Equation (5.27) is the extra factor q^{-d} because of the integer part $d \cdot T_s$ of the time-delay. The zero due to the fractional part L of the time-delay is the same as before (Equation (5.29)).

The sampled version of a proper rational continuous-time system with m zeros and n poles has generically $n-1$ discrete-time zeros, some of which may go to infinity or be canceled by poles [ÅSH84]. The sampled version of a continuous-time system can have zeros due to three reasons:

1. Zeros of the continuous-time system [ÅSH84].
2. Zeros created by the sampling process for a continuous-time system without fractional time-delay. Some of these zeros may be "unstable", i.e. be outside the unit circle. Whether they are outside the unit circle, depends on $n-m$ and the sampling interval T_s. If $n-m<2$ or the sampling interval is long enough, these zeros are always inside the unit circle. See [ÅSH84].
3. Zeros created by the sampling process for a continuous-time system with fractional time-delay [FMS91, ÅW84].

In [FMS91] it is said that "sampling zeros", of type 2 above, does not significantly affect the result of the estimation. This conclusion is based upon a simple example.

5.3.2 Recursive TIDEA

In [FMS91] the fact that $\alpha \to L/T_s = \epsilon$ when $T_s \to 0$ is utilized. We call ϵ the *normalized fractional time-delay.* Using this we see that the zero z_0 (Equation (5.29)) of the system in Equation (5.27) can be written

$$z_0 = -\frac{\alpha}{(1-\alpha)} \approx -\frac{\epsilon}{1-\epsilon}$$

for small sampling intervals. We can solve for ϵ in this expression:

$$\epsilon = \frac{1}{1-z_0}.$$

In [FMS91] an extended model structure, compared to Equation (5.30) is employed:

$$G(q) = q^{-(d+1)} \frac{b_2 q^2 + b_1 q + b_0}{A_R(q)}, \tag{5.31}$$

where $A_R(q)$ is a polynomial in q with a high enough degree.

This model structure (due to the numerator $b_2 q^2 + b_1 q + b_0$) can handle normalized fractional time-delays $-1 < \epsilon < 1$, not only $0 < \epsilon < 1$ as in Section 5.3.1, without changing the normalized integer part d of the time-delay.

Also, higher order true systems than a first order system as in Section 5.3.1 can be modeled thanks to the denominator $A_R(q)$. Requirements for this is that the true continuous-time system does not have any zero on the imaginary axis and that the sampling interval is small enough. If the true time-delay is wanted, the polynomial $A_R(q)$ should have a high enough degree for $1/A_R(q)$ to accurately describe the rational part $\bar{G}_1(s)$ of the time-delay system. On the other hand, if the role of the time-delay is to adjust the phase of the model in an important frequency range, $A_R(q)$ can have a low degree. Compare with Section 1.1.

In [FMS91] time-delay estimation method, called *TIDEA* (TIme Delay Estimation Algorithm), based on the above, is presented. The model (5.31) is estimated recursively. TIDEA also contains a special mechanism for adjusting the normalized integer part d of the time-delay. See [FMS91] for details. According to [FMS91], the global convergence of TIDEA is still not proved. We did not succeed to implement the method without it diverging. Therefore we have no simulation results for this method.

5.3.3 Exact time-delay from the sampling process

The zero-order-hold sampling of the model structure (5.26) is described in Section 5.3.1. Assume that all model parameters except for the time-delay are known. From Equations (5.25), (5.28) and (5.29) it is easy to solve for the fractional time-delay L:

$$L = \frac{1}{\beta} \ln \left(\frac{1 - e^{\beta T_s} z_0}{1 - z_0} \right) \tag{5.32}$$

Note, that this expression does not require that $T_s \to 0$ as in TIDEA (Section 5.3.2). The integer part of the time-delay must be estimated in some other way, e.g. as in TIDEA or with `oestruc` (Section 5.2.2).

In [ÅSH84] the system

$$\bar{G}(s) = \frac{1}{s} e^{sL}$$

is zoh sampled, giving

$$G(p) = \frac{(T_s - L)q + L}{q(q-1)}$$

with the zero

$$z_0 = -\frac{L}{T_s - L}. \tag{5.33}$$

By solving for the fractional time-delay L in Equation (5.33), we get

$$L = \frac{z_0 T_s}{z_0 - 1}.$$

6

Area, Moment and Higher-Order Statistics Methods

Area and moment methods (Sections 6.1-6.3) utilize relations between the time-delay and certain areas above or below the step response $s(t)$ and certain moments of the impulse response $h(t)$ (integrals of the type $\int t^n h(t)dt$). *Higher-order statistics (HOS) methods* (Section 6.4) use HOS to discriminate between symmetric probability distribution, e.g. Gaussian, and non-symmetric. They are are suitable when desired and undesired signals differ in this aspect.

6.1 Area Methods

In [ÅH95] some methods for identifying simple process models of the kind

$$G(s) = \frac{K}{1 + sT} e^{sL} \tag{6.1}$$

and

$$G(s) = \frac{K}{(1 - sT)^2} e^{sL} \tag{6.2}$$

are described. Part of this identification is the estimation of the time-delay L, which is what we are interested in here.

Area methods [ÅH95] use area calculations of a step response $s(t)$. First the *average residence time* T_{ar} is calculated as

$$T_{\text{ar}} = \frac{A_0}{K},$$

where $K = s(\infty)$ is the static gain and the area A_0 is calculated from

$$A_0 = \int_0^\infty (s(\infty) - s(t))dt. \tag{6.3}$$

The average residence time is a rough measure of the time for the step response to finish. Then we calculate

$$A_1 = \int_0^{T_{\mathrm{ar}}} s(t)dt. \tag{6.4}$$

For the first order model (6.1) the time constant is then given by

$$T = \frac{e^1 A_1}{K}. \tag{6.5}$$

The time-delay is

$$L = T_{\mathrm{ar}} - T.$$

For the second order model (6.2) the time constant is given by

$$T = \frac{e^2 A_1}{4K} \tag{6.6}$$

and the time-delay is

$$L = T_{\mathrm{ar}} - 2T.$$

See [ÅH95] for the derivation of these area methods. In [ÅH95] they use a real step response from a step response experiment when identifying the system with these methods. In this thesis we will use an arbitrary input signal and first estimate the step response before applying the methods.

6.2 Moment Methods

In moment methods [ÅH95] we interpret the normalized impulse response

$$f(t) = \frac{h(t)}{\int_0^\infty h(t)dt}$$

of the system as a probability density function. The quantity $h(t)$ is the impulse response. Then we can call the quantities

$$m_n = \int_0^\infty t^n f(t)dt \tag{6.7}$$

the moment of order n. For the first order model (6.1) the static gain is given by

$$K = \int_0^\infty h(t)dt.$$

The average residence time will be

$$T_{\text{ar}} = m_1 \tag{6.8}$$

and the time constant is solved from

$$T^2 = m_2 - T_{\text{ar}}^2.$$

The time-delay is as for the corresponding area method:

$$L = T_{\text{ar}} - T.$$

For the second order model (6.2) K and T_{ar} is calculated as for the first order model. The time constant is solved from

$$T^2 = \frac{1}{2}m_2 - \frac{1}{2}T_{\text{ar}}^2$$

and the time-delay given by

$$L = T_{\text{ar}} - 2T.$$

See [ÅH95] for the derivation of these moment methods. In [ÅH95] these methods require an impulse response experiment to be performed. In this thesis we will use arbitrary input signals and then estimate the impulse response before employing the moment methods above.

In [ÅH95] an other method is described that, by using moments of the input $u(t)$ and output signals $y(t)$, can be used with any signal that decays quickly enough. A variant of this method is also described that can give an approximative model when using input and output signals that do not decay but are bounded by $e^{\alpha t}$ for large t, where α is a user-chosen parameter which influence the accuracy of the approximation.

6.3 An Area and Moment Method with Better Noise Properties

In [Ing03] it is noticed that in the calculation of moments, like Equation (6.7), the values of $h(t)$, $u(t)$ or $y(t)$ at late times in the integration will have a higher weight t^n. This will give a rapidly decreasing signal to noise ratio (SNR) for late times, since the noise level is about constant for all times but the signal is decaying. This results in inaccurate estimates of the moments. This is probably a major reason for the poor performance of the moment methods, described in Section 8.6.

In [Ing03] a modified area and moment method for the first order system (6.1) is described. By defining the signals $y_d(t) = \frac{d}{dt}y(t)$ and $u_d(t) = \frac{d}{dt}u(t)$ the factor t in the integral m_1 (6.7) in Equation (6.8) disappears giving integrals of only $u(t)$ and $y(t)$ for the calculation of T_{ar}. By computing the time constant T from Equation (6.5) in the area method for the first order system, also the second order moment integral is avoided. The identification data are collected by a combination of a setpoint change experiment in closed loop and a step response experiment in open loop.

This method could be called a combined area and moment method. It could also be called a pure area method because that is what the result is.

6.4 Higher-Order Statistics Methods

For the signal model

$$\begin{aligned} x(t) &= u(t) + n_1(t) \\ y(t) &= u(t - T_d) + n_2(t) \end{aligned} \tag{6.9}$$

(problem 2b in Section 2.1) Nikias and Pan suggest in [NP88] time-delay methods employing higher-order statistics (third order cumulants and cross cumulants and bispectrum and cross bispectrum). Conditions are, however, that the interesting signal $u(t)$ has a non-symmetric probability distribution (Gaussian not allowed) and the noises $n_1(t)$ and $n_2(t)$ have symmetric probability distributions, e.g. Gaussian. The noises can be temporally and spatially correlated. Correlation based methods have problems with spatially correlated noise [Mat98, NM93].

Assume that $u(t)$ is a zero mean, stationary random process with non-zero skewness (non-symmetric probability distribution). Assume that $n_1(t)$ and $n_2(t)$ are zero mean, Gaussian, stationary random processes. $n_1(t)$ and $n_2(t)$ may be correlated but they are independent of $u(t)$. With the signal model (6.9) and the above assumptions we have the following relations [NM93, NP88]:

$$R_{xxx}(\tau, \rho) = R_{uuu}(\tau, \rho) \tag{6.10}$$

$$R_{xyx}(\tau, \rho) = R_{uuu}(\tau - T_d, \rho), \tag{6.11}$$

where the 3^{rd} order (cross) moment is defined $R_{xyz}(\tau, \rho) = \mathrm{E}\left\{x(t)y(t+\tau)z(t+\rho)\right\}$. The time-delay can be found from these equations by finding a maximum in the 2D time domain (τ, ρ) or study the phase in the frequency domain. By 2D Fourier transform of Equations (6.10)-(6.11) we get the (cross) bispectra

$$\begin{aligned} \Phi_{xxx}(\omega_1, \omega_2) &= \Phi_{uuu}(\omega_1, \omega_2) \\ \Phi_{xyx}(\omega_1, \omega_2) &= \Phi_{uuu}(\omega_1, \omega_2) \cdot e^{i\omega_1 \Delta t}. \end{aligned}$$

These two types of methods (2D time domain and 2D frequency domain) are similar to the two subclasses *time domain approximation methods* and *frequency domain approximation methods* of the class *time-delay approximation methods*. Perhaps also the HOS method class should be divided in the two subclasses *2D time domain HOS* and *2D frequency domain HOS methods*.

Part II

Comparison and properties of time-delay estimation methods

7

Simulation Setup and Analysis

Simulations have been conducted in MATLAB in order to experimentally study time-delay estimation (TDE) methods. Both open loop and closed loop simulations were executed. The circumstances or *factors* were slightly different for the open loop and closed loop simulations. First in this chapter, a short presentation of some approaches to perform experiments are given. Second, a short description of the implementation of the investigated time-delay estimation methods is given in Section 7.2. Then the open loop simulations are described in Section 7.3 and after that the closed loop simulations in Section 7.4. Finally, analysis methods are described in Section 7.5.

7.1 Factorial Experiments

Assume that we have a system (not necessarily a dynamic system) or process that we want to study and draw conclusions about. We want to know how one or several *factors* influences the output (the result) of the system. Each factor has two or more possible *levels* (values). Often we want to know the factor level combination that in some sense gives the best result. When performing experiments with the aim to draw conclusions about the system, several approaches are possible:

In the *best-guess approach* one factor at a time is changed while the other factors are kept at their previous levels. This approach works reasonable well if the experimenter already has knowledge and experience of the system. There are, however, some drawbacks: It can take a long time to find an acceptable combination of factor levels and it is not certain that the best combination is found.

In the *one-factor-at-a-time approach* one factor at a time is varied over its possible range while the other factors are kept at base-line levels. This approach has some drawbacks. The main drawback is that it does not discover interactions between factors. Interactions are common and when they occur this approach usually gives poor results. This approach is also less efficient (needs more observations) than some other approaches.

In the *factorial approach* the factors are varied together instead of one at a time. Advantages with this approach are: 1) It can discover interactions between factors and thereby avoid misleading conclusions. 2) It is more efficient (needs fewer observations) than approaches where only one factor is changed at a time. 3) Conclusions of a factor can be drawn that are valid over a range of levels of the other factors. In this thesis the factorial approach, giving *factorial experiments*, is used.

The material of this section has been taken from [Mon97].

7.2 Implementation of Estimation Methods

Here, a short description is given of the implementation of the time-delay estimation methods used in simulations in this thesis.

First, time domain approximation methods (Section 4.1) are listed. The methods idimp4, idstep4, idimpCusum3, idstepCusum3, idimp5, idstep5, idimpCusum4 and idstepCusum4 are implementations of direct and CUSUM thresholding of estimated impulse and step responses (Algorithm 2). Implementation details can be found in [Bjö03b].

idimp4, idstep4. These method threshold directly the impulse and step response estimates (Section 4.1.2 and 4.1.6 and Algorithm 2), respectively. The number of estimated coefficients is 70. The thresholds are $h(t) = h_{\text{std}} \cdot \hat{y}_{\text{std}}(t)$, where h_{std} is a user selected constant and $\hat{y}_{\text{std}}(t)$ is the estimated standard deviation of the impulse or step response estimate, respectively. If none of the estimated coefficients reach above $h(t)$ then a "missed detection" is recorded.

idimp5, idstep5. Since h_{std} (Algorithm 2) in idimp4 and idstep4 are difficult to choose manually to suit all input signal types and SNRs, they have been chosen by a simulation study (Section 8.1.1 and reference [Bjö03b]) to $h_{\text{std}} = 5$ for both idimp5 and idstep5. This would give a confidence interval with a confidence level of 100.0% if <the residuals> the impulse or step response estimates are Gaussian distributed (which is a good assumption, see [Lju99]) and the estimate of the standard deviation of the impulse or step response estimate is accurate. The input signal was prewhitened (Section 7.3) for idimp5 even if it turns out that does not matter (Section 8.1.1). idstep5 do not use prewhitening.

idimpCusum3, idstepCusum3. For low SNR the estimated impulse and step responses are very noisy (Figure 4.3 and [Bjö03b, KG81]). In an attempt to mitigate this the methods idimpCusum3 and idstepCusum3 use CUSUM (cumulative sum) thresholding (Algorithm 2), which is a nonlinear averaging operation (Section 4.1.3). The user-selected parameters in CUSUM are the relative drift ν_{std} and the relative threshold h_{std}. The used (absolute) drift and threshold are $\nu = \nu_{\text{std}} \cdot \hat{y}_{\text{std}}(0)$ and $h = h_{\text{std}} \cdot \hat{y}_{\text{std}}(0)$.

idimpCusum4, idstepCusum4. Because the relative drift ν_{std} and threshold h_{std} are difficult to select manually, they have also been chosen by a simulation study [Bjö03b] to $\nu_{\text{std}} = 1$ & $h_{\text{std}} = 3$ for idimpCusum4 and to $\nu_{\text{std}} = 6$ & $h_{\text{std}} = 1$ for idstepCusum4. The input signal was prewhitened for idimpCusum4, even if this does not have a large effect [Bjö03b]. idstepCusum4 does not use prewhitening.

kurz. Another approach to the thresholding is employed in [KG81] and its implementation is here called kurz.

Frequency domain approximation methods (Section 4.2):

lagudap. The DAP method (Section 4.2.5) applied to a Laguerre model (Section 4.2.6) with 10 terms in the sum ($n_{\text{lag}} = 10$) and the Laguerre pole $\alpha = 0.8$.

firdap. The DAP method applied to a FIR model $y(t) = B(q)u(t) + w(t)$ with 15 taps (the order of the $B(q)$ polynomial).

arxdap. The DAP method applied to an ARX model $y(t) = (B(q)/A(q))u(t) + (1/A(q))w(t)$ with the order of the $A(q)$ polynomial $n_a = 4$, the order of the $B(q)$ polynomial $n_b = 15$ and the number of time-delays $n_k = 0$.

oedap. The DAP method applied to an OE model $y(t) = (B(q)/F(q))u(t) + w(t)$ with the order of the $B(q)$ polynomial $n_b = 15$, the order of $F(q)$ polynomial $n_f = 2$ and the number of time-delays $n_k = 0$.

lagucont1. See Section 4.2.4. Laguerre model (Section 4.2.6) with 10 terms in the sum ($n_{\text{lag}} = 10$) and the Laguerre pole $\alpha = 0.8$. No zero guarding .

lagudap2. The DAP method (Section 4.2.5) applied to a Laguerre model (Section 4.2.6) with 10 terms in the sum ($n_{\text{lag}} = 10$) and the Laguerre pole $\alpha = 0.8$ with zero guarding (Section 4.2.7) with ZType=+1, ZSize=0.15 and ZNo=3 (Section 8.2.1). No prewhitening (Section 7.3).

firdap2, arxdap2, oedap2. As lagudap2 but with replacing the Laguerre model with a FIR (15 taps), ARX ($n_a = 4$, $n_b = 15$) or output error ($n_f = 2$, $n_b = 15$) model [Lju99].

Laguerre domain approximation methods (Section 4.3):

fischer1, fischer2, fischer3, fischer4, fischer5, fischer6, fischer7, fischer8. See Section 4.3.3. Table 7.1 contains the used parameters. For fischer1-fischer4 the pole was chosen to $\alpha = 0.6$ or $\alpha = 0.8$ (see [Bjö03e]). For fischer5-fischer8 the pole position 0.995 was selected according to [Fis99, FM99b, FM99c]: "by minimizing

Table 7.1 Parameters of the implemented Laguerre domain approximation methods. The number of used Laguerre functions is $N + 1$. "# sing.vals" means the number of retained singular values in Algorithm 3 (page 41).

Name	$N + 1$	Algorithm	Pole α	# sing.vals
fischer1	51	tausvd	0.8	5
fischer2	51	taulp1	0.8	5
fischer3	150	tausvd	0.6	5
fischer4	150	taulp1	0.6	5
fischer5	51	tausvd	0.955	5
fischer6	51	taulp1	0.955	5
fischer7	150	tausvd	0.955	5
fischer8	150	taulp1	0.955	5

the squared equation error between the input signal and its approximation by a truncated ($N = 16$) Laguerre series with respect to α". The algorithms tausvd and taulp1 (Algorithm 3 and 4) were employed.

Continuous-time one-step explicit methods (Section 5.1). These are here also called *idproc methods.*

idproc1, idproc2, idproc3, idproc4, idproc5. Methods using simple process models (Section 5.1.1) and with/without prewhitening of the input signal (Section 7.3) . An upper limit of the estimates is 30.

idproc6, idproc7. Use the first order and second order process models (Section 5.1.1), respectively. An upper limit of the estimates is 30. No prewhitening of the input signal (Section 7.3) was used.

Discrete-time one-step explicit methods (Section 5.2):

arxstruc, oestruc, met1struc. These methods are described in Sections 5.2.1, 5.2.2 and 5.2.3 respectively. The model orders (Section 5.2) and with/without prewhitening (Section 7.3). could be chosen.

oestruc3, arxstruc3, met1struc3. The methods arxstruc, oestruc, met1struc with model orders [Lju99], ($n_f = 2$, $n_b = 1$), ($n_a = 10$, $n_b = 5$) and ($n_a = 10$, $n_b = 1$) and without prewhitening the input signal (chosen in Section 5.2).

Two-step explicit methods (Section 3):

elnaggar. This is a recursive discrete-time two-step method [EDE89].

Area and moment methods (Sections 6.1-6.2):

area1, moment1. The area and moment methods in Sections 6.1-6.2 on estimated step and impulse responses (Section 4.1.2). Parameters to choose were with/without prewhitening of the input signal, two model structures (Section 6.1), and two ways to perform the numerical integration. See [Bjö03b] for details of the implementation.

area2, moment2. The area and moment methods using the first order model (Sections 6.1-6.2) without prewhitening the input signal and certain ways to perform the numerical integration. See [Bjö03b] for implementation details.

As part of the work underlying this thesis is a MATLAB function estdeadtime which implements some of the the time-delay estimation methods above.

7.3 Open Loop Simulation Setup

In the open loop simulations the output signal $y(t)$ was simulated as

$$y(t) = G(s)u(t) + v(t) \tag{7.1}$$

where $u(t)$ and $v(t)$ are the input and noise signals respectively. $G(s)$ is a linear system with time-delay.

The *factors* (or conditions) of the simulations can be divided into *fixed factors* and *varied factors*. The different possible choices for the same factor are in this thesis called *levels* of the factor. The choice of level for all factors is called a *factor level combination*. A *trial* consists of a collection of simulations, one simulation for each possible combination of factor levels. In factorial experiments (Section 7.1), a trial is called a *replicate*.

Fixed factors for the open loop simulations were: The noise $v(t)$ was white and Gaussian. The system $G(s)$ was simulated by the function `lsim` in [CST] in continuous-time. The sampling interval for the time-delay estimation was $T_s = 1$. The varied factors for the open loop simulations are given below:

Method. The time-delay estimation methods. See Section 7.2.

Prewhite. For many of the methods in Section 7.2 the input and output signals $u(t)$ and $y(t)$ were either

pw. Filtered through a filter that made the input white. See Algorithm 10.

nopw. Not filtered through a prewhitening filter.

Sys. The following systems were simulated (see Figure 7.1 for impulse responses):

slow2. A slow second order system (same as G_2 in Equation 4.28 in Section 4.2.6):

$$G_1(s) = e^{-9s}\frac{1}{(10s+1)(s+1)} \tag{7.2}$$

Algorithm 10 Prewhitening the input signal.

1. Remove the mean value of the input $u(t)$ and output $y(t)$ signals.
2. Estimate a 10^{th} order AR model for $u(t)$.
3. Filter $u(t)$ and $y(t)$ through the AR model.

fast2. A fast second order system:

$$G_2(s) = e^{-9s} \frac{1}{(s+1)(0.1s+1)} \tag{7.3}$$

real4. A fourth order system with real poles (*real4*). The poles and zeros lie between the poles of system G_1:

$$G_5(s) = e^{-9s} \frac{0.05(s+0.9)(s+0.4)}{(s+1)(s+0.6)(s+0.3)(s+0.1)} \tag{7.4}$$

cplx4. A fourth order system with complex poles. The poles have the same distance to the origin as in system G_1:

$$G_6(s) = e^{-9s} \frac{1/36(s+0.9)(s+0.4)}{(s-0.1\cdot 2^{-1/2}(-1\pm i))(s-2^{-1/2}(-1\pm i))} \tag{7.5}$$

As can be seen in Figures 7.1 this system is very slow and the oscillations are weak.

G_7. A slow first order system:

$$G_7(s) = e^{-9s} \frac{1}{(10s+1)} \tag{7.6}$$

G_8. A fast first order system:

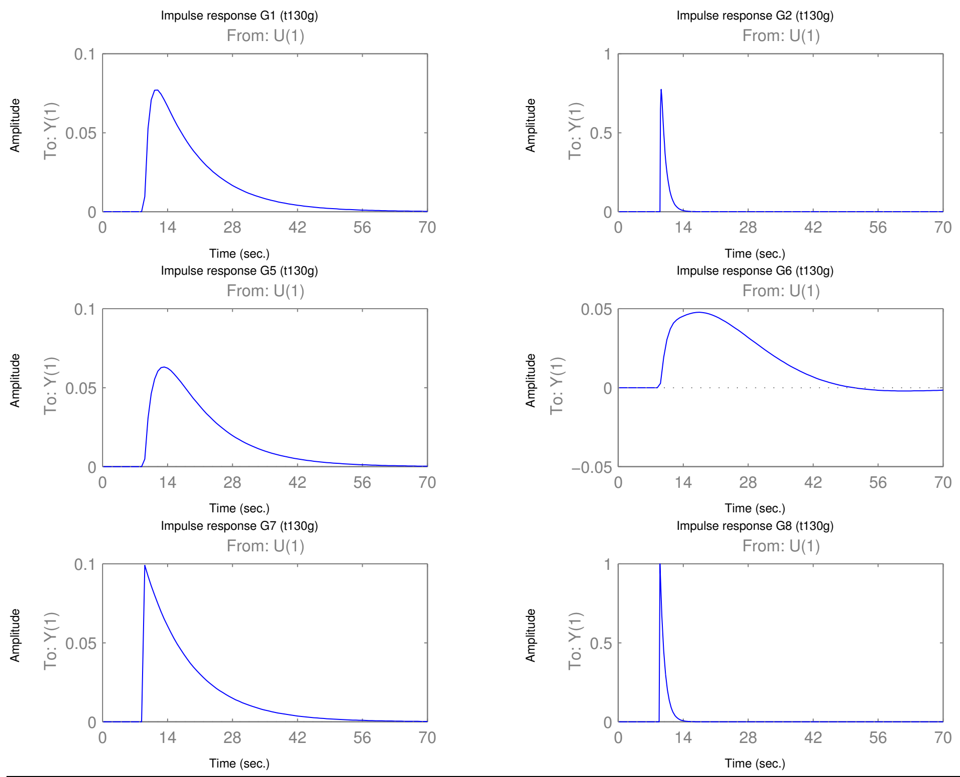
$$G_8(s) = e^{-9s} \frac{1}{(s+1)} \tag{7.7}$$

G_9. A pure time-delay system:

$$G_9(s) = e^{-9s} \tag{7.8}$$

Note that for all the systems above, the time-delay will be 10 after zero-order-hold sampling because a time-delay of one sample is created by the sampling.

Figure 7.1 Impulse response of system G_1-G_2 and G_5-G_8 (Equations 7.2-7.7). (t130g1)



InType. The input signal was 500 samples long and could be of three different types:

RBS 10-30%. (Random Binary Signal) with frequency contents between 10%-30% of the Nyquist frequency. It was generated by the function `idinput` in [SIT]. The same realization of the signal was used in all trials of the same Monte Carlo simulation. See Figure 7.2 for an example of this input signal type.

RBS 0-100%. RBS with frequency contents between 0%-100% of the Nyquist frequency, i.e white noise. It was generated by the function `idinput` in [SIT]. The same realization of the signal was used in all trials of the same Monte Carlo simulation. See Figure 7.3 for an example of this input signal type.

Steps. Three steps of the form (in MATLAB code): `[zeros(50,1);ones(150,1); -ones(150,1); zeros(150,1)]`. See Figure 7.4 for an example of this input signal type.

Figure 7.2 Time signal (left) and frequency spectrum (right) for a realization of the input signal type RBS 10-30%. (t160a1time, freq)

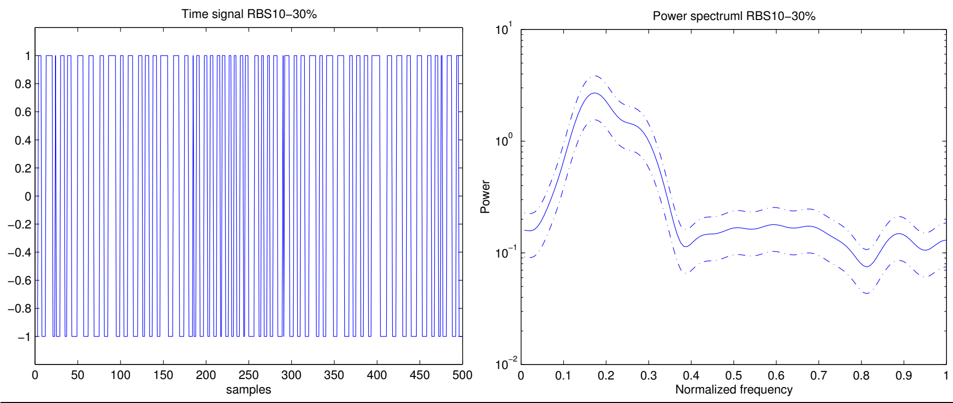
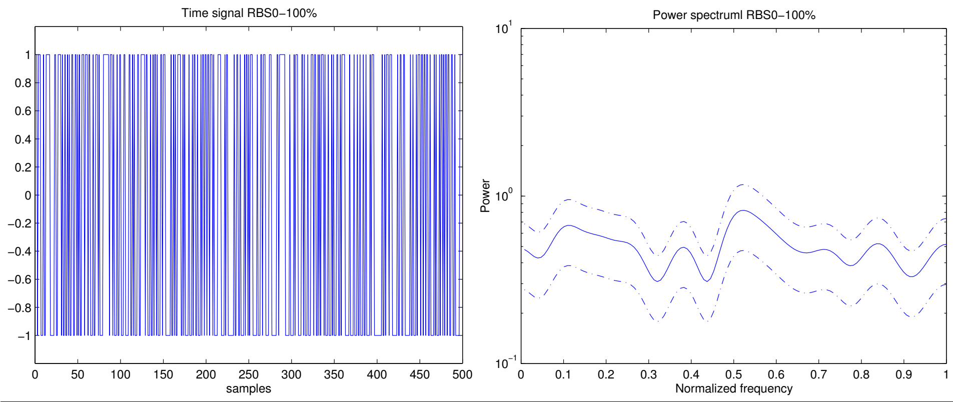


Figure 7.3 Time signal (left) and frequency spectrum (right) for a realization of the input signal type RBS 0-100%. (t160a2time, freq)



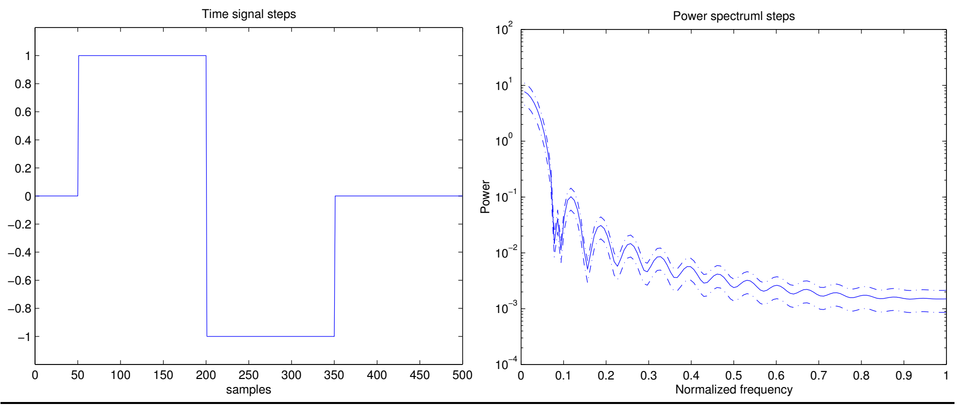
SNR. The signal-to-noise ratio (*SNR*) could be chosen. The SNR was defined as the ratio of the output signal power (without noise) to the output noise power. The used SNR has been either 1 or 100.

Many of the factors and levels of the open loop simulations were chosen to be the same as in [Hor00], which makes comparisons easy. The names of the factors and factor levels in this section are also used in the following plots in this thesis.

7.3.1 A standard benchmark

In many simulations in this thesis a standard setup (*the standard benchmark*) was used. It consists of calculating the average RMS error, bias or standard deviation of the estimates for the systems G_1-G_2, G_5-G_6 (not including G_7-G_9), for

Figure 7.4 Time signal (left) and frequency spectrum (right) for a realization of the input signal type Steps. (t160a3time, freq)



the input signals RBS 10-30%, RBS 0-100% and Steps and for the SNRs 100 and 1. This simulation setup was the same as in [Bjö03e]. It is also the same as in [Bjö02, Bjö03b, Bjö03f, Bjö03c, Bjö03d] but with a different definition of the SNR (See [Bjö03e]).

Note that the results with this simulation setup are valid for the simulated setup and the used methods. For example, since there are three slow systems and one fast system, the result will be more adapted to slow systems if taking the average over the systems.

7.4 Closed Loop Simulation Setup

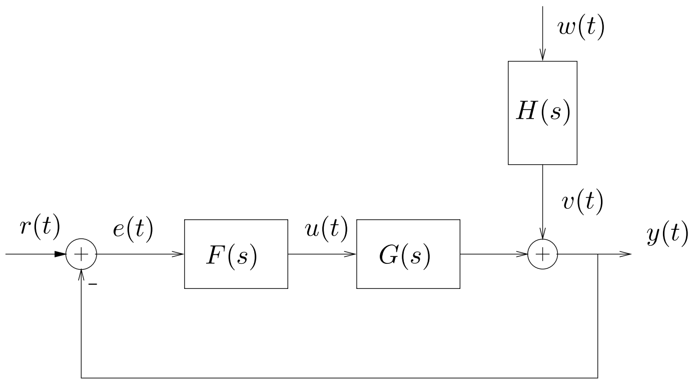
In automatic control of systems and processes, usually feedback is used, resulting in closed loop systems. Therefore it is interesting to evaluate time-delay estimation methods in closed loop. This section describes the used setup for closed-loop simulations. The control system of the closed loop simulations is depicted in Figure 7.5.

Fixed factors for the closed loop simulations were: The noise source $w(t)$ was white and Gaussian. The continuous-time system $G(s)$ was converted to discrete-time with zero-order-hold sampling before the simulations. The simulations were performed in discrete time. The sampling interval was 1.

Most of the factors in Section 7.3 could also be varied in the closed loop case but sometimes the levels were different. Only one input signal type was used so the factor InType was not varied. Extra factors were Ctrl, NoiseMod and InputPlace. Many of the factors and levels of the closed loop simulations were chosen to be the same as in [IHD00, IHD01a, IHD01b], which makes comparisons easy.

Sys. The following systems were simulated. Again, the time-delay will be 10 after the sampling.

Figure 7.5 The control system in closed loop simulations. $F(s)$ is the controller, $G(s)$ the system with the time-delay and $H(s)$ is the noise model.



slow. A slow second order system:

$$G_{1b}(s) = 0.5e^{-9s}\frac{1}{(10s+1)(s+1)} \tag{7.9}$$

This is the same as system $G_1(s)$ in Section 7.3 except for the static gain being half of $G_1(s)$'s gain.

fast. A fast second order system:

$$G_{2b}(s) = 0.5e^{-9s}\frac{1}{(s+1)(0.1s+1)} \tag{7.10}$$

This is the same as system $G_2(s)$ in Section 7.3 except for the static gain being half of $G_2(s)$'s gain.

Ctrl. The possible controllers were all PI-controllers but giving different bandwidths. T_s is the sampling interval.

slow. A slow PI controller:

$$F_1(q) = 0.6\frac{1 + T_s/50 - q^{-1}}{1 - q^{-1}} \tag{7.11}$$

medium. A medium fast PI controller:

$$F_2(q) = 0.25\frac{1 + T_s/5 - q^{-1}}{1 - q^{-1}} \tag{7.12}$$

fast. A fast PI controller:

$$F_3(q) = 0.9\frac{1 + T_s/12 - q^{-1}}{1 - q^{-1}} \tag{7.13}$$

NoiseMod. The noise model $H(q)$ could be chosen in three different ways:

filt. White noise filtered by

$$H_1(q) = \frac{0.71}{1 - 0.7q^{-1}} \tag{7.14}$$

oe. Unfiltered white noise giving an output error model structure $y(t) = G(q)u(t) + w(t)$:

$$H_2(q) = 1 \tag{7.15}$$

arx. A noise model $H_3(q)$ with the same denominator as system G_{1b}. This would give an ARX model structure $y(t) = (B(q)/A(q))u(t) + (1/A(q))w(t)$ with G_{1b} except for a different normalization in the numerator of $H_3(q)$:

$$H_3(q) = \frac{0.92}{1 - 0.368q^{-1} + 1.671 \cdot 10^{-5}q^{-2}} \tag{7.16}$$

InType. In the closed loop simulations, only one reference signal $r(t)$ was used. It is here called *Steps2* and consisted of two steps and 2000 samples and was generated in MATLAB by: `[4*ones(1,1000), 2*ones(1,1000)].'`. This signal is not exactly the same as in [IHD01b] but of a similar character.

InputPlace. The input to the time-delay estimation could be either

u. The input signal $u(t)$ to the system and the output signal $y(t)$ or

r. The reference signal $r(t)$ and the output signal $y(t)$.

SNR. In the simulations the power of the noise $w(t)$ was specified to 0.10, which is the same as the high noise power used in [IHD00, IHD01b]. This means that the SNR according to our definition in Section 7.3 will be different for different systems and controllers. In a simulation the SNR was estimated. The maximum SNR was 1147, the minimum 790, the mean 959 and the standard deviation 66. In summary, this is a very high SNR.

7.5 Analysis Methods

In this work we have used the following analysis methods. However, the result of not all analysis methods are shown in this thesis. There are also some special analysis methods that are specific to certain estimation methods. See the respective section about these.

1. Plots of the estimates. These give the possibility to discover outliers and see the behavior for different factor level combinations.

2. Bar plots of the RMS (root mean square) error of the estimates for different factor level combinations. These give an estimation quality measure which can be used to see how good the estimates are. They also indicate which factor level combinations, e.g. methods, are better or worse than others. See Section 7.5.2.

3. Bar plots of the bias or standard deviation of the estimates for different factor level combinations. See Section 7.5.2.

4. ANOVA (ANalysis Of VAriance) [Mon97, Mat01] on the RMS errors for different factor level combinations. ANOVA can tell us if there is a statistically significant difference between factor level combinations, e.g. different methods or systems. See Section 7.5.3.

5. Plot of confidence intervals for pair-wise comparisons [Mon97, Mat01] for the RMS errors for different factor level combinations. This can give us the possibility to say that certain factor level combinations, e.g. some methods, are better than others. See Section 7.5.3.

RMS error, bias and standard deviation are expressed in number of sampling intervals in this thesis. A part of the work underlying this thesis is the development of a MATLAB toolbox for managing and analyzing data from factorial experiments [Bjö]. It uses some functions from the MATLAB STATISTICS TOOLBOX [Mat01].

7.5.1 Managing failing estimates

Many methods sometimes fail and return estimates with unreasonable values (negative, large positive, complex or NaN). To handle this, the following procedures were used:

1. In this procedure, complex and NaN values and values ≤ 0 or ≥ 20 are replaced with the value 20. This will give a large error for these estimates as the true time-delay is 10. The maximum RMS error will also be 10. Then, the 90%, 95% or 100% best estimates are retained. Estimates of the factor level combinations with the worst estimates will be removed first. If some factor level combination contain only failed estimates (RMS error =10) then the results in the plots will be the same for 90%, 95% and 100% as long as not all estimates are removed from any factor level combination. This is the case for the methods idimp5, idstep5 and idstepCusum4 in Figure 9.1. On the contrary, if the failed estimates are spread uniformly between factor level combinations, the difference between 90%, 95% and 100% will be large, as with idproc7 in Figure 9.1.

2. In this procedure, estimates outside the region [0,20] are set to the value 20. This choice is a trade-off between how much to punish failed estimates and how much to punish poor RMS error values when the estimates occurred.

It is up to the user to decide what is important. The closer this number ($\hat{T}_d = 20$) is to the correct time-delay (true $T_d = 10$) the less punishment of the method for failed estimates. The more distant from correct time-delay the more punishment for failed estimates. This can mean that depending on the chosen punishment for failures different methods will be "best". The time-delay $\hat{T}_d = 20$ is assumed to be a maximum time-delay to detect.

3. In this procedure the optimization is restricted to the region [0, 30].

The motivation for removing the worst estimates is that a good implementation of the estimation method can detect time-delay estimates outside an expected range ([0,20] in procedure 1 above) and should be able to handle this, e.g. by restarting an optimization with a different initial value.

7.5.2 Plots of RMS error, bias and standard deviation

The computational steps for bar graphs with RMS error, bias and standard deviation of time-delay estimates are:

1. Optionally, remove the worst estimates. See Section 7.5.1
2. Compute the RMS error, bias or standard deviation of the time-delay estimates for each factor level combination separately.
3. Compute the average (or max) of the result from step 2 over the levels of the factors which are not to be shown in the plot.
4. Plot the graph.

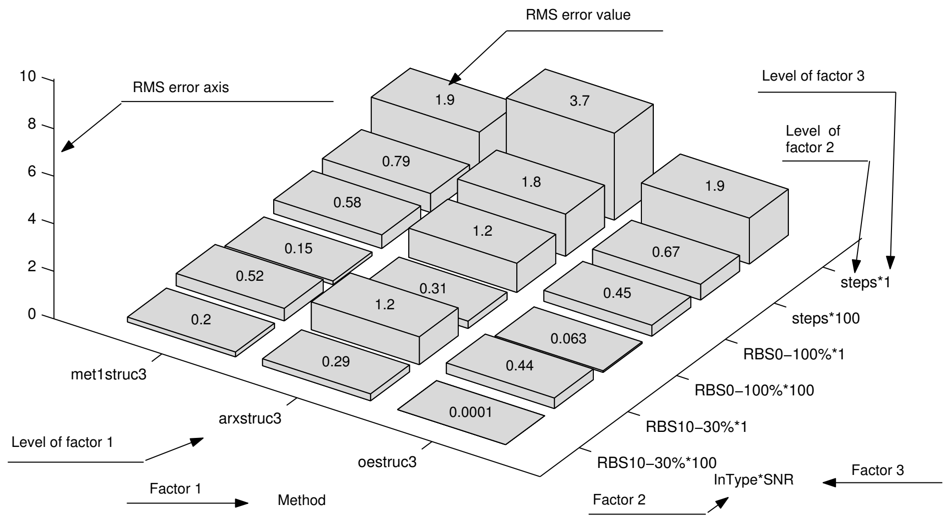
Thus, all results in the bar graphs are on average (or max) over all the factors that are not explicitly studied. Certain special cases can give a different result. Also keep in mind that the presented RMS values only are estimates of the "true" ones.

Figure 7.6 shows an example plot. The axis label "InType*SNR" means that on this axis there are different (factor level) combinations of input signal type and SNR. On the other axis there are different methods. The tick mark labels tell us what factor level combinations there are in the different "rows" and "columns". The levels of the factors are separated by an asterisk "*". For example, on the axis to the right "steps*1" means a combination of the input signal type Steps (Section 7.3) and SNR=1. The level 10-30% is an abbreviation of RBS 10-30% and 0-100% of RBS 0-100% . See Sections 7.3-7.4 for definitions of the factor and level names.

7.5.3 ANOVA and confidence intervals

A statistical method for analysis of experiments used in some scientific disciplines is ANOVA (ANalysis Of VAriance) with subsequent computation of confidence intervals for pair-wise comparisons, see for example [Mon97]. These analysis methods make it possible to give statements and conclusions with a certain level of confidence or a specified risk of being false.

Figure 7.6 Example of plot of RMS error of time-delay estimates as a function of methods and environment factors. (t208b13)



In ANOVA the following linear statistical model is used (here for just two factors):

$$x_{ijk} = \mu + \tau_i + \beta_j + (\tau\beta)_{ij} + \epsilon_{ijk}, \tag{7.17}$$

where x_{ijk} is the *response variable* (the observation), μ is the *overall mean effect*, τ_i is the *main effect* of the i^{th} level of the first factor, β_j is the *main effect* of the j^{th} level of the second factor, $(\tau\beta)_{ij}$ is the ij^{th} *interaction effect* of the first and second factors. All these effects are assumed to be constant and model (7.17) is called the *fixed effects model.* The quantity ϵ_{ijk} is a random error. Moreover, the following is assumed: $\sum_i \tau_i = 0$, $\sum_j \beta_j = 0$, $\sum_i (\tau\beta)_{ij} = 0$ and $\sum_j (\tau\beta)_{ij} = 0$. The errors ϵ_{ijk} should be independent and Gaussian distributed with zero mean and constant variance [Mon97]. Constant variance means that the variance of ϵ_{ijk} is the same for all ijk. In the ANOVA, hypothesis tests are performed to test all $\tau_i = 0$, all $\beta_j = 0$ and all $(\tau\beta)_{ij} = 0$ against not all $\tau_i = 0$, not all $\beta_j = 0$ and not all $(\tau\beta)_{ij} = 0$, respectively.

In this thesis, we have utilized ANOVA and confidence intervals to discover significant differences between method parameters (method factors) and environment parameters (environment factors).

The trials were split into four groups. Each group was used to compute an estimate of the RMS error of the time-delay estimate, giving four estimates $\tilde{x}_{ijk}$, $k = 1, \ldots, 4$, of the RMS error for each factor level combination ij. Here only quantities for two factors (with indices i and j) are shown. In the simulations and analyses often more factors have been used. When forming the RMS error, a large quantity of independent random numbers with the same probability distribution are added. The well-known central limit theorem [Mon97] says that this sum will be asymptotically Gaussian distributed. This makes it possible to use ANOVA and confidence intervals for our problems if the number of trials is large enough.

Each RMS error estimate $\tilde{x}_{ijk}$ was then transformed by

$$x_{ijk} = \tilde{x}_{ijk}^{\lambda}$$

[Mon97, p. 84-90], giving us the the response variable x_{ijk}, $k = 1, \ldots, 4$. The four groups of trials have resulted in four replicates x_{ijk}, $k = 1, \ldots, 4$. The reason for the transformation was to (try to) fulfill one of the requirement of ANOVA, namely the constant variance of ϵ_{ijk} in Equation (7.17). This transformation is useful in cases when the variance of the observation increases with the size of the observation. It is also useful when the data has a non-Gaussian, skewed, probability distribution, because in skewed distributions the variance often is a function of the mean value. See [Mon97]. If the standard deviation $\sigma_{\tilde{x}}$ of a random variable $\tilde{x}$ is proportional to a power of the mean value μ of $\tilde{x}$: $\sigma_{\tilde{x}} \propto \mu^{\alpha}$ and we transform $\tilde{x}$ by $x = \tilde{x}^{\lambda}$, it can be shown that the standard deviation of the transformed variable is $\sigma_x \propto \mu^{\lambda+\alpha-1}$ [Mon97]. Thus, by letting $\lambda = 1 - \alpha$, the variance of the transformed variable will become constant regardless of its mean value.

Figure 7.7 shows an example transformation. The transformation is $x = \tilde{x}^{0.96}$. The horizontal axis is the logarithm of the mean of the four RMS estimates and the vertical axis is the logarithm of the standard deviation of the four RMS estimates for all factor level combinations. The transformation is chosen by fitting a straight line to the data points by the least squares method. Outliers whose values on the vertical axis are very low are first removed. These outliers appear when the estimation is so good that it always give the correct answer and therefore the standard deviation is zero. They also appear when the estimates always are very poor and are replaced with a fixed value, as with procedure 2 in Section 7.5.1. Before the logarithm is computed, the value `eps`, which is the smallest number that can be represented in the computer ($\approx 10^{-16}$ in the used computer type, SUN SunBlade 100), is added to avoid the logarithm of zero. This will, in our case, give a value on the vertical axis of about -16 for outliers and can be seen in Figures A.1 and A.4.

After the transformation, the four replicates x_{ijk} of the response variable can be applied to ANOVA, which can be used to discover significant differences between levels of factors and level combinations of factor interactions. With the aid of ANOVA, we can either say that there are statistically significant differences between the levels or we can say nothing. If there are differences, we cannot say which level is the "best". We need the confidence intervals for that. The ANOVA results in the traditional ANOVA table. See Table 7.2 for an example. The table tells us that there are significant differences between the main effects of methods, between the main effects of systems and between combinations of methods and systems (the column "Prob>F" contains very small values). We say that these factors and interactions have *(statistical) effect.* The values in this column ("Prob>F"), also

Figure 7.7 Example of plot (without transform) for choosing a variance-stabilizing transform [Mon97] for ANOVA. The transformation became (RMS error)$^{0.96}$. The transformation is chosen by fitting a straight line to the data points by the least squares method. (t224b1)

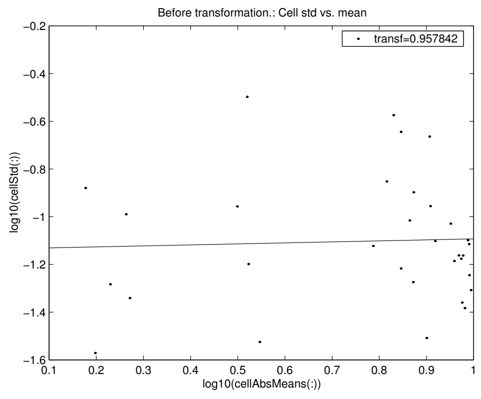


Table 7.2 Example of ANOVA table. Constrained (Type III) sums of squares [Mat01]. (t224b1)

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
Method	2769.8838	7	395.6977	9274.3874	0
Sys	84.5622	3	28.1874	660.6579	0
Method*Sys	115.7668	21	5.5127	129.2071	0
Error	4.0959	96	0.042666		
Total	2974.3087	127			

commonly called *p-values*, are the risks of saying that there is a difference when there is actually no difference.

For the factors and interactions which have effect it is meaningful to compute and plot confidence intervals for pair-wise comparisons of factor levels or level combinations of interactions. Figure 7.8 shows an example. If the confidence intervals (the horizontal lines in the circles) are not overlapping there is a significant difference between the factor levels and we can see which level is the "best". If the transformation is positive ($\lambda > 0$), the intervals to the left in the figure are better than those to the right. "Better" means that they correspond to a lower value of the response variable, which is here the RMS error. If the transformation is negative ($\lambda < 0$), it is the opposite, namely the best intervals are to the right. If the intervals are overlapping we cannot say which level, if any, is the best. The quantity on the horizontal axis is the response variable x which is the transformed RMS error: $x = \tilde{x}^{\lambda} = (\text{RMS error})^{\lambda}$. All confidence interval plots in this thesis has a simultaneous confidence level of 95%. This means that the probability that all confidence intervals cover their respective parameter is 95% or with other words the risk that we incorrectly say that there a difference between any of factor levels or level combinations of interactions is 5%. Note that the horizontal lines in the plots are not confidence intervals for $(\mu\tau)_i$, see below.

For the reader interested in statistical details we will describe the confidence interval plot in Figure 7.8 in another way. We start by writing the ANOVA model (7.17) in three other forms, namely

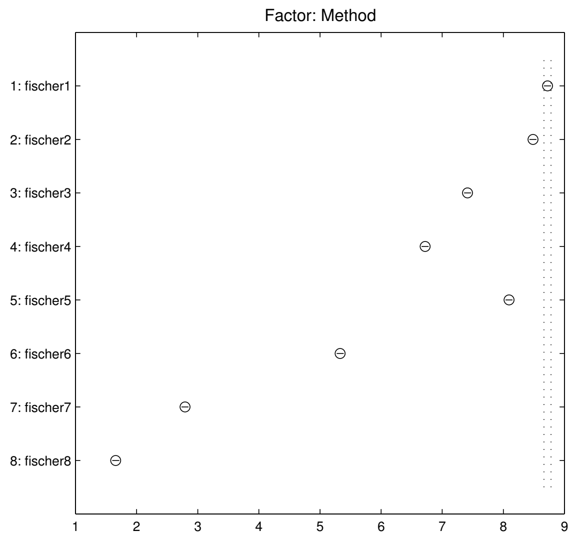
$$x_{ijk} = (\mu\tau)_i + \beta_j + (\tau\beta)_{ij} + \epsilon_{ijk} \quad \text{with} \quad (\mu\tau)_i = \mu + \tau_i \tag{7.18}$$

$$x_{ijk} = \tau_i + (\mu\beta)_j + (\tau\beta)_{ij} + \epsilon_{ijk} \quad \text{with} \quad (\mu\beta)_j = \mu + \beta_j \tag{7.19}$$

$$x_{ijk} = \mu_{ij} + \epsilon_{ijk} \quad \text{with} \quad \mu_{ij} = \mu + \tau_i + \beta_j + (\tau\beta)_{ij} \ , \tag{7.20}$$

where $(\mu\tau)_i$ is the *main mean* of the i^{th} level of the first factor, $(\mu\beta)_j$ is the *main mean* of the j^{th} level of the second factor, μ_{ij} is here called the ij^{th} *interaction mean* of the first and second factors (actually, μ_{ij} is the mean for the combination of the i^{th} level of the first factor and the j^{th} level of the second factor in the complete model (7.17) where the interaction are included). We want to compare the "means" above to see which factor levels or factor level combinations that give the best or worst result (highest or lowest RMS error of the time-delay estimate). This can be done by computing confidence intervals for differences in the means. For example, let us assume that the interactions are negligible. If the confidence interval for $(\mu\tau)_n - (\mu\tau)_m$, where n and m correspond to different levels of the first factor, does not include the value zero or negative values, then the level n gives a significant higher value of the response variable (=RMS error if no transformation is used) than the level m. Comparing all possible combinations of levels with each other will often give very many comparisons. An economical way to perform all these comparisons is utilized in the confidence interval plots in this thesis. The circles in these plots (see Figure 7.8) mark the values of $\widehat{(\mu\tau)}_i$, i.e. estimates of $(\mu\tau)_i$. The horizontal lines in the circles have half the length of the confidence intervals for $(\mu\tau)_n - (\mu\tau)_m$. The confidence interval for $(\mu\tau)_n - (\mu\tau)_m$ does not include the value zero exactly when the horizontal lines from neighboring circles do not overlap. The horizontal lines are not restricted to be within the circles but can sometimes extent outside the circles as in Figure 8.10. The reason for the lines being so short in this thesis is that very many trials are used in the simulations. As already said, the horizontal lines in the plots are not confidence intervals for $(\mu\tau)_i$ but $1/2$ of the confidence intervals for $(\mu\tau)_n - (\mu\tau)_m$ (with the simultaneous confidence level of 95%). The confidence intervals in this thesis use Tukey's honestly significant difference criterion [Mat01]. When choosing parameters of time-delay estimation methods, we can ourselves select the factor levels (the parameter values) and we will study confidence intervals for $(\mu\tau)_n - (\mu\tau)_m$ if the interactions are significant. When studying environment factors, which we cannot choose ourselves, we are often only interested in how the time-delay estimation methods perform on average and study confidence intervals only for main means even if there are interactions that have a statistical effect. This is somewhat unusual.

Figure 7.8 Example of a confidence interval plot (95% simultaneous confidence level). Tukey's honestly significant difference criterion [Mat01] is used. The quantity on the horizontal axis is the transformed RMS error: (RMS error)$^{\lambda}$. (t224b1)



To examine if the requirements of ANOVA are fulfilled it is customary to study some plots of the *residuals*

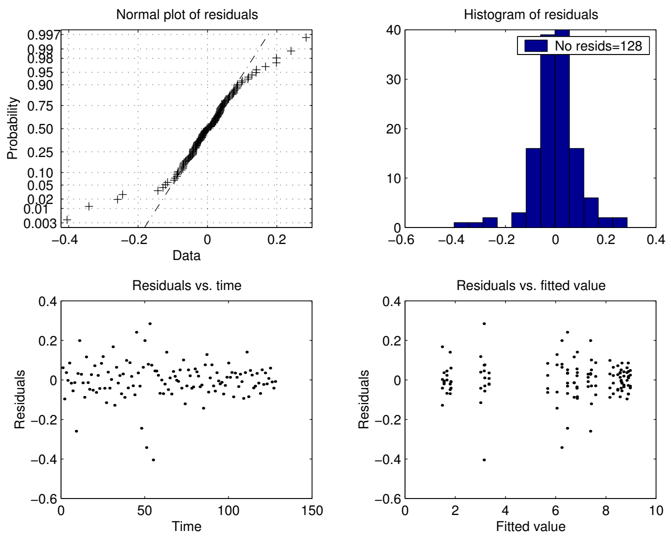
$$\varepsilon_{ijk} = x_{ijk} - \hat{x}_{ijk}, \tag{7.21}$$

where $\hat{x}_{ijk}$ is an estimate of the observation x_{ijk}. The quantity $\hat{x}_{ijk}$ is also called the *fitted value.* Figures 7.9-7.10 show examples of such plots. The residuals should be Gaussian distributed (which implies that also the response variable for a fixed factor level combination is Gaussian). This can be checked by the two top graphs in Figure 7.9. The data points in the top left graph should follow the dash dotted straight line. "In visualizing the the straight line, place more emphasis on the central values than on the extremes." [Mon97] The top right plot should resemble the well-known Gaussian bell. In Figure 7.9 these graphs look good. Figure A.2 is an example when these graphs do not look good.

The residuals vs. time and vs. fitted values should be within horizontal bands and be "structureless" (show no obvious patterns) [Mon97], which can be checked by the two bottom graphs in Figure 7.9. In Figure 7.9 these graphs look fine. In Figure A.2 these graphs have a clear structure which is not good. If the residuals increase with increasing fitted value, this is a sign of that the variance is not constant ([Mon97]).

In Figure 7.10 the requirement that the variance of ϵ_{ijk} is constant can be checked. In this figure the standard deviation of the residuals ε_{ijk} are displayed for

Figure 7.9 Example of residual analysis for ANOVA and confidence intervals. (t224b1)

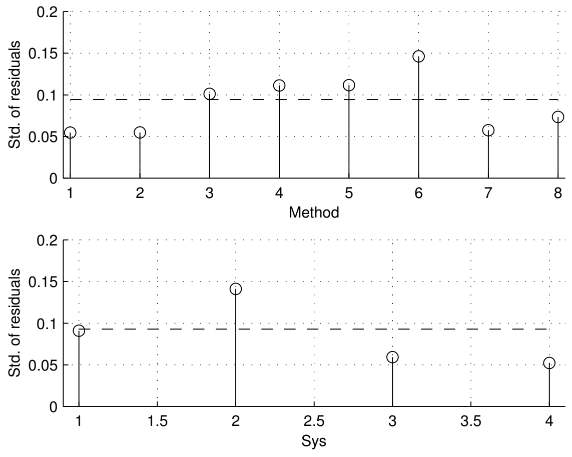


different levels of all factors. If the model is exact then $\varepsilon_{ijk} = \epsilon_{ijk}$. The standard deviation should be equal for all levels. In Figure 7.10 the difference in standard deviation is a factor two. In Figure A.3 all factors have approximately constant variance except for the threshold (bottom left graph). The middle value of the threshold has a much higher variance than the other values.

If the condition of Gaussian residuals is not met, then the length of the confidence intervals will not be correct and the confidence level of the ANOVA hypothesis tests will not be the advertised. Moreover, if the variance is different for different factors levels or factor level combinations, the confidence intervals should have different lengths. In this thesis the average variance is used in the confidence interval plots and therefore all intervals is of an average length. If any of these two conditions, Gaussian residuals and constant variance, are violated the interpretation of the results must be done with caution.

The residuals from TDE methods are not always Gaussian distributed. Especially the time-delay estimation methods that deliver discrete values (multiples of the sampling interval) are difficult to get a Gaussian response variable from. Both the facts that they deliver discrete values and that the delivered values often are the same (often either completely correct or completely incorrect) require the sum of very many squared errors to get something that resembles a Gaussian distribution. The time-delay estimates (and therefore also the squared errors) are very non-Gaussian. If many trials are needed, the advantage of ANOVA and confidence intervals is smaller.

Figure 7.10 Example plots of residual standard deviation versus factor levels. (t224b1)



However, since the results in the ANOVA tables and the confidence interval plots often are very clear (the confidence intervals are well separated), we have some security margin for non-constant variance and non-Gaussian distribution to avoid drawing false conclusions. See for example Figures 8.1, A.2 and A.3, where the validation graphs (Figures A.2 and A.3) do not look good but there are two clearly separated groups of confidence intervals in Figure 8.1. We say that the confidence intervals in different groups are significantly different but we do not say that confidence intervals within the same group are significantly different even if the intervals do not overlap.

Another remedy is that the ANOVA method for the *balanced* (equal number of replicates for all factor level combinations) *fixed* (constant) *effects model* (7.17), is rather robust against violations of the Gaussianity condition [Mon97, p. 82] and the constant variance condition [Mon97, p. 85]. We also hope for this robustness when the prerequisites for ANOVA are not fully fulfilled. We also trust the random number generator of Matlab to give us independent experiments and response variables.

It is also important to keep in mind, that even if the ANOVA and confidence intervals tell us that there is a significant difference between some levels or level combinations, it is not necessary that these differences are important in practice. For example, assume that we can show that two car models have statistically different fuel consumptions. But if the difference is only 0.00001 liter per kilometer, it probably has no practical importance.

This section has explained ANOVA and confidence intervals for pair-wise comparisons: what they do and what requirements must be fulfilled to use them. We will use these analysis methods in the subsequent chapters.

8

Parameters and Properties of Time-Delay Estimation Methods

In this Chapter, parameters in time-delay estimation (TDE) methods are chosen by analysis of Monte Carlo simulations to optimize their estimation quality. Also, some properties of different classes of TDE methods are observed. The sections in this chapter follow the classification given in Chapter 3.

8.1 Time Domain Approximation Methods

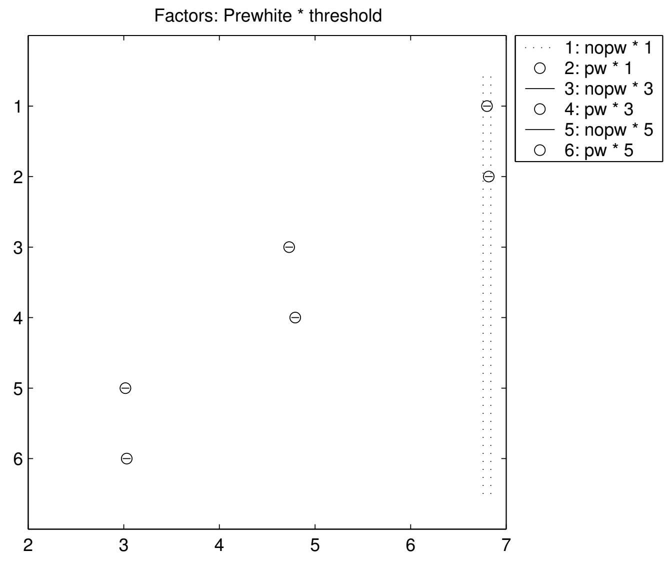
In this section we will choose parameters and study properties of the time domain approximation methods idimp4, idstep4, idimpCusum3 and idstepCusum3 (Section 7.2). These methods can also be called *thresholding methods*.

8.1.1 Choosing parameters in thresholding methods

Here we will show how we can choose the the relative threshold h_{std} and with/without prewhitening in the method idimp4 in Section 7.2. Some simple experiments showed that it is difficult to manually select method parameters that are suitable for all cases (all level combinations of the environment factors input signal type and SNR). Therefore, the parameters are chosen with the help of confidence intervals. Often the methods miss to detect and fail to deliver an estimate [Bjö03b]). In such cases procedure 2 in Section 7.5.1 is employed.

In the ANOVA table, Table A.1, the p-value (Section 7.5.3) for prewhitening is 0.053, which is a rather low risk for saying that there is a difference when there is no. However, since the prerequisites of the ANOVA (Figures A.2A.3) seem not to be fulfilled we cannot say that there is any difference between with and without

Figure 8.1 Confidence intervals (95% simultaneous confidence level) for combinations of method parameters for thresholding of impulse response idimp4. Positive transformation: (RMS error)$^{0.91}$=> "The lower the better". (t228b4)



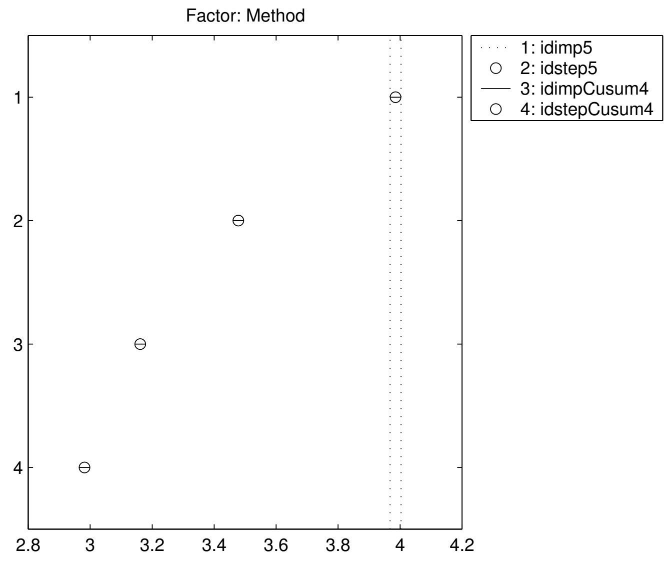
prewhitening. On the other hand, we see in Table A.1 that there is a significant difference for different thresholds because the p-value for this factor is very low. Figure 8.1 depicts confidence intervals for different parameter combinations. We see from Figure 8.1 that there are two choices (to the left in the figure) of method parameters which are significantly better on average than the other but with no significant difference between them. One of these is $h_{\text{std}} = 5$ with prewhitening (number 6 in the graph). This choice was called idimp5 in Section 7.2. As said, the prerequisites seem not to be fulfilled but since the results are very clear (well separated intervals) in Figure 8.1 we trust this conclusion.

The parameters of the time domain estimation methods idstep4, idimpCusum3 and idstepCusum3 (Section 7.2) are chosen in the same way in [Bjö03b], giving the methods idstep5, idimpCusum4 and idstepCusum4 (Section 7.2).

8.1.2 Properties of thresholding methods

The standard benchmark simulation setup in Section 7.3.1 was employed in this section. When a method missed to detect, a uniform distributed random number in the range 20 to 30 was delivered as the time-delay estimate. The reason for using random numbers was to come closer to the required prerequisites for the confidence interval calculations (Section 7.5.3). Since the estimates are very non-Gaussian, as many as 4096 trials were simulated. The transformation was (RMS error)$^{0.89}$

Figure 8.2 Confidence intervals for pair-wise comparisons (95% simultaneous confidence level) for different thresholding methods. Positive transformation: (RMS error)$^{0.89}$=> "The lower the better". (t229b5.m)



(Section 7.5.3). This means "The lower the better". We see in the validation graphs (Figures A.5-A.6) that the prerequisites are not completely fulfilled so we must be somewhat careful in the interpretation of the ANOVA and the confidence intervals.

Figure 8.2 shows confidence intervals for pair-wise comparisons of the methods idimp5, idstep5, idimpCusum4 and idstepCusum4. We see that:

- Step response is significantly better (not overlapping confidence intervals) than impulse response on average.
- CUSUM thresholding is significantly better than direct thresholding on average.

In [Bjö03d] it is shown that the thresholding methods idimp5, idstep5, idimpCusum4 and idstepCusum4 give better estimates for RBS signals than step signals and for fast systems than slow systems.

It is also shown in [Bjö03d] that the methods overestimate the time-delay. Nearly the whole RMS error is caused by the positive bias. The reason is that the detection miss due to a too high threshold. With the relative threshold $h_{\text{std}} = 5$, the absolute threshold will be very high. If the estimated impulse response coefficients are Gaussian distributed $N(y(t), y_{\text{std}}(t))$, then $\hat{y}(t) \pm 5y_{\text{std}}(t)$ covers the true impulse response coefficient $y(t)$ by a probability of 100.0%. This is the same as wanting to be very certain not to say that the impulse response has started when it has not. This also means that the start of many true impulse responses will be missed. The true confidence level will often, however, be different from this (100.0%) because the confidence intervals will be uncertain since 1) often the model and the true systems are of different structures 2) the standard deviation must be estimated and 3) the true probability distribution is not always Gaussian.

The used thresholds have nevertheless been selected to give the best result on average (Section 8.1.1). Different relative thresholds are needed in different combinations of environment factors. In Figure 9.2 it can be seen that these thresholding methods give very accurate estimates for some combinations of input signal type and SNR but very poor for other. A better estimation of the change time than simple thresholding is needed. The references [CHWW99, Isa97, KG81] contain suggestions for improvements.

Probably it would be better to estimate the start of the impulse response by going backwards from the detection time than using the detection time. See [Gus00] for how this can be done for CUSUM thresholding. Compare with Section 4.1.3 and Figure 4.2.

8.2 Frequency Domain Approximation Methods

In this section we will choose parameters and study properties of frequency domain approximation methods (Section 4.2). These methods can also be called *phase methods*.

8.2.1 DAP in open loop

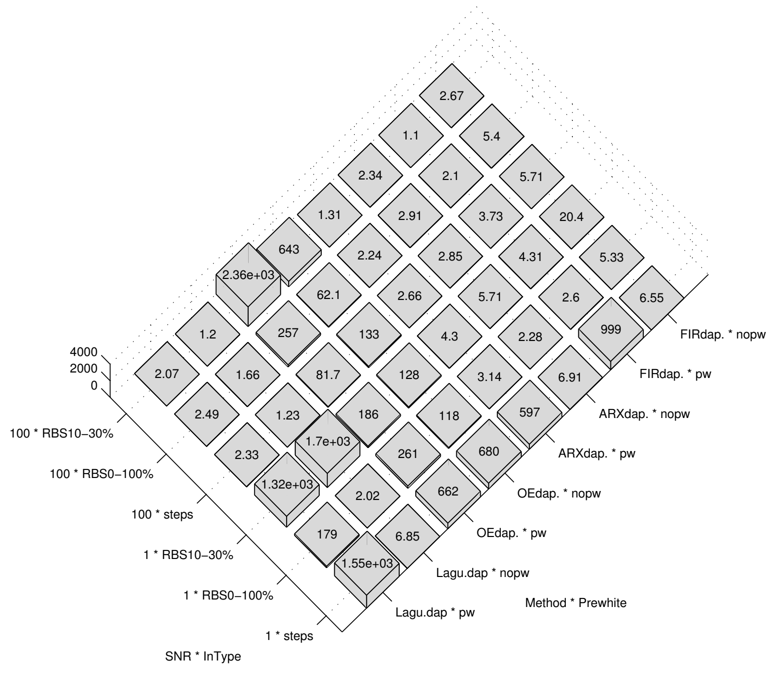
Figure 8.3 depicts the RMS error of the DAP methods (Section 4.2.5) lagudap, oedap, arxdap and firdap (Section 7.2) with and without prewhitening (Section 7.3) in the standard benchmark (Section 7.3.1) using 1024 trials. For each level combination of the other factors, the system with the worst, i.e. highest, RMS error was plotted. In this way the worst case performance is optimized. Which systems that were chosen is not shown in the figure.

As can be seen in Figure 8.3, some of the DAP methods exhibit very large RMS errors in some cases. The methods totally fail for these factor level combinations. We notice that some estimates are very large in these cases, up to about 30000 (not shown here). Compare with Section 4.2.7.

In Figure 8.3, lagudap without prewhitening seems to be the best method for the input signal type Steps at high SNR. It is also among the best methods for low SNR. The method oedap fails in all cases. As a summary of Figure 8.3 we can say that the DAP methods seem to be non-robust and they can totally fail in some cases.

In order to mitigate the failures of the DAP methods, the use of zero guarding according to Section 4.2.7 on the Laguerre model structure was investigated in [Bjö02]. Two different types of zero guarding (ZType) were investigated: Remove zeros outside the unit circle close to the point +1 or remove zeros outside

Figure 8.3 RMS error of DAP methods (Sections 4.2.5 and 7.2) in the standard benchmark (Section 7.3.1). 1024 trials. No zero guarding. For each combination of the other factors, the worst system (with the largest RMS error) is plotted. Axis labels and tick mark labels are explained in Section 7.5.2. (t230b1)



but close to the unit circle (ucirc). Different distances (ZSize) from +1 or the unit circle to be considered as "close" were tried: 0.05, 0.10, 0.15, 0.20, 0.25, 0.25 and 0.30. If there were several zeros within the zero guarding distance, different maximum number (ZNo) of zeros to remove were tested: 1 to 4. The closest zeros were removed first. A simulation with 1024 trials was conducted. The three input signals types and the four systems in the standard benchmark (Section 7.3.1) were used. No prewhitening was employed. The SNR was low [Bjö02] because we want the estimation method with zero guarding to manage low SNRs. ANOVA and confidence intervals (Section 7.5.3) were utilized to discover significant differences between the different combinations of the levels of ZType, ZSize and ZNo. The result in [Bjö02] showed that there are several "good" combinations of ZType, ZSize and ZNo with no significant differences. It is, however, clear that removing zeros close to +1 is better than removing zeros close to the unit circle (ucirc). It is also clear that allowing for removing more than one zero is necessary. We choose one of the good combinations in [Bjö02]: ZType=+1, ZSize=0.15 and ZNo=3 for the next simulation we will present.

Figure 8.4 RMS error of DAP methods. 1024 trials. The zero guarding ZType=+1, ZNo=3 and ZSize=0.15. For each combination of the other factors, the worst system (with the largest RMS error) is plotted. Axis labels and tick mark labels are explained in Section 7.5.2. Compare with Figure 8.3. (t232b1)

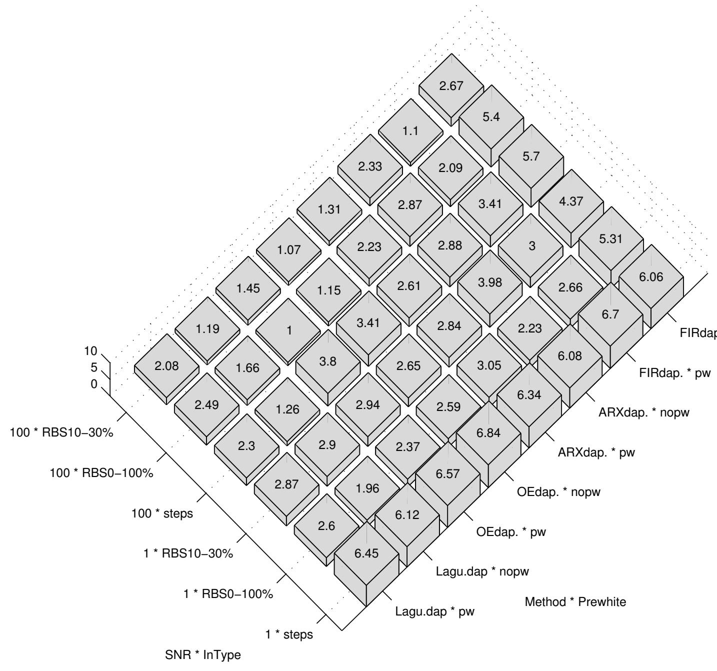
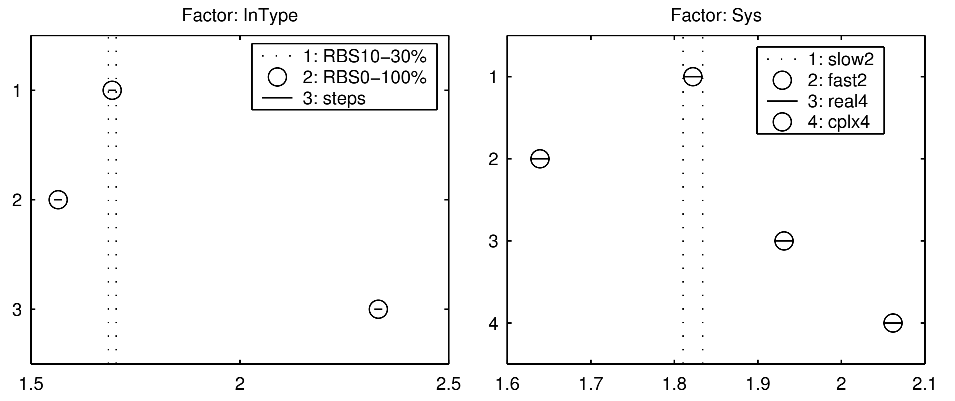


Figure 8.4 shows the result for RMS error when zero guarding above was used, cf. Figure 8.3. We immediately observe that now no DAP methods fail but give reasonable and low RMS errors. Even oedap works and gives good estimates. Still lagudap gives the best result for Steps at high SNR. At low SNR it is among the best. The “optimum” choice of ZType, ZNo and ZSize for lagudap appears to work also for the other DAP methods. In closed loop simulations [Bjö02] the same zero guarding as for open loop appeared to work.

8.2.2 More properties of DAP methods

In this section another simulation is used. The four methods lagudap2, firdap2, arxdap2 and oedap2 (Section 7.2) were put into service for low SNR (=1) in the standard benchmark (Section 7.3.1). The worst estimates were removed by procedure 1 in Section 7.5.1. The transformation (Section 7.5.3) became positive: (RMS error)$^{0.64}$. This means the lower value in the confidence interval plot the better. ANOVA table and validation graphs can be found in Appendix A.2. We see in Figure 8.14 that

Figure 8.5 Confidence intervals for pair-wise comparisons (95% simultaneous confidence level) of input signal types (left) and systems (right) for the average over lagudap2, firdap2, arxdap2 and oedap3 (Section 7.2). Positive transformation: (RMS error)$^{0.64}$=> "The lower the better". (t208b16)



- On average these methods work better for fast than slow systems.
- On average these methods work better for random binary input signals than step input signals.

8.2.3 Discussion

Of the tested model structures, FIR, ARX and Laguerre models can be quickly estimated by linear regression. The OE model structure, on the other hand, requires a numerical search which makes the estimation slower.

The location α of the pole of the Laguerre model (Section 4.2.3) probably affects the locations of the zeros and therefore also how sensitive they are for falling on the wrong side of the unit circle, i.e. here outside it (Section 4.2.6), due to noise.

As seen in this thesis, the DAP method fails if a zero erroneously falls outside the unit circle. The error in the time-delay estimation is not proportional to the error in the position of the zero. A small error in the position which does not cause the zero to fall on the wrong side of the unit circle will result in a small estimation error. But when the zero just falls on the incorrect side of the unit circle the estimation error will be large. The estimation error is larger close to the point $+1$ and smaller further away. This makes the DAP method sensitive to where the true zero is located.

An advantage of the DAP method compared to many other methods is that it can estimate subsample time-delays, i.e. time-delays that are a fraction of a sampling interval.

The zero guarding chosen in Section 8.2.1 for Laguerre in open loop seems to work also for closed loop. It also appears to function for the other model structures in both open loop and closed loop. Thus, the choice of zero guarding appears to be robust.

In Section 8.2.1 we saw that sometimes it is not enough to remove only one zero. The reason is likely that because complex zeros come in complex conjugated pairs, both zeros in the pair must be removed.

The location of the true non-minimum phase zeros probably depends on the time-delay and the sampling interval. The longer time-delay or the shorter sampling interval, the closer to the point +1 will the true zeros be. Therefore ZSize depends on the maximum possible time-delay and on the sampling interval.

A reason why Laguerre DAP works well for step input signals could be as follows. It is well known that an estimated model will become better in frequency ranges where the input signal has much energy [Lju99]. Since a step-like input signal has its most energy at the frequencies which are used by the time-delay estimation, i.e at low frequencies, it will result in a good time-delay estimate.

In the open-loop simulations the fast second order system gave the lowest RMS error on average. This is natural because this system exhibits a clearer start of the rise in the step response. The fourth order system with complex poles gave the highest RMS error on average. It seems that a more complex system makes the time-delay estimation more difficult.

For step signals in open and closed loop, we in this thesis agree with [Hor00, IHD01b] that Laguerre DAP is a suitable estimation method. [Hor00, IHD01b] have, however, only tested with step signals and not reported that the DAP method sometimes fails.

8.2.4 Conclusions

We draw the following conclusions from our work on DAP methods:

- The DAP method is not restricted to the Laguerre model structure but can be used with any linear model structure, e.g. OE, ARX or FIR.
- DAP methods are inherently non-robust and can totally fail in some cases.
- The probability of failure of a DAP method depends on the model structure of the estimated model, the input signal type and the SNR.
- In failing cases, DAP methods can be made more robust by zero guarding. A means to choose the zero guarding for a certain application is using confidence intervals on estimates from simulated signals.
- An appropriate choice of zero guarding appears to be robust and work for several model structures in both open loop and closed loop.
- For zero guarding of Laguerre DAP, removing zeros outside the unit circle close to +1 works better than removing zeros at other places outside but close to the unit circle. We must also allow more than one zero (a complex conjugated pair) to be removed.

- Most often the DAP methods work better without prewhitening the data.
- OE DAP requires much more computation time than FIR, ARX or Laguerre DAP. See also Section 9.4.

8.3 Laguerre Domain Approximation Methods

In this section we will choose parameters and study properties of Laguerre domain approximation methods (Section 4.3).

8.3.1 The standard benchmark

We use the standard simulation setup in Section 7.3.1. Before the analysis methods are employed, outliers and failed estimates have been removed by procedure 1 in Section 7.5.1 .

Figure 8.6 displays the RMS error for combinations of method and input signal type. By comparing Figure 8.6 with other RMS error plots in this thesis, e.g. Figure 9.1, the following conclusions are drawn:

- Step input signals are possible to use but RBS signals are not.
- For step input signals, the method fischer8 seems to be the best.

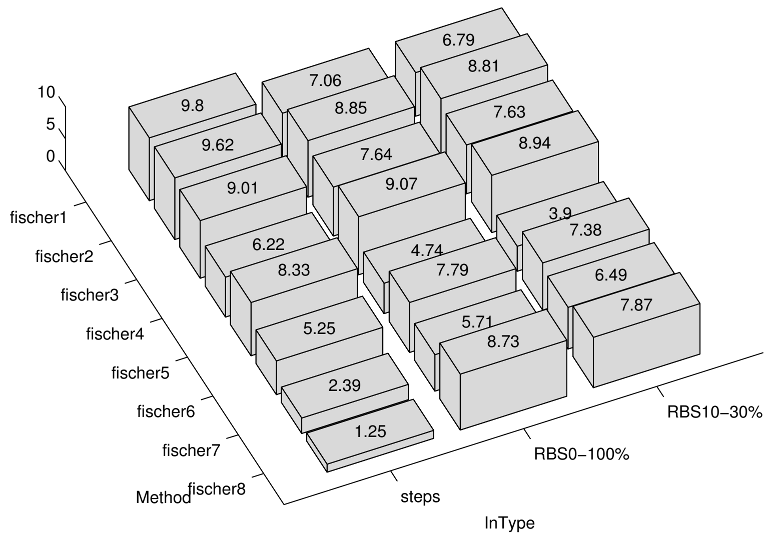
8.3.2 Several Laguerre domain approximation methods with step input

We now restrict ourselves to step signals and compare the different Laguerre domain approximation methods with the aid of confidence intervals for pair-wise comparisons.

We use the estimates for the low SNR (SNR=1). The number of trials was 1152. Outliers and failed estimates have been removed by procedure 1 in Section 7.5.1. The ANOVA table [Bjö03e] shows that there are significant differences between combinations of methods and systems. The confidence interval plot in Figure 8.7 shows that fischer8 indeed is significantly better than the other methods for step input. The validation graphs for the use of ANOVA and the confidence interval plot in Appendix A.3.1 look very fine.

When using high SNR (SNR=100), the prerequisites for the ANOVA and the confidence interval plot were not fulfilled (determined by the same type of graphs as in Appendix A.3.1). Since the case with low SNR is a more difficult case than with high SNR, we consider it to be more informative. It must be more difficult for a method to be best for low SNR. It should then not be bad for high SNR either.

Figure 8.6 Average RMS error (in number of sampling intervals) for different Laguerre domain approximation methods and input signal types for the standard benchmark. Axis labels and tick mark labels are explained in Section 7.5.2. (t224b1)

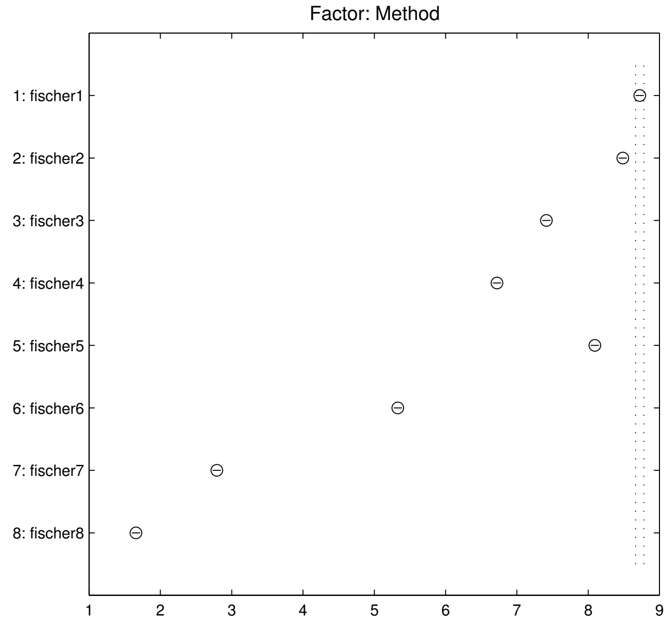


8.3.3 The method fischer8 with step input signals

We now concentrate on fischer8 and step input signals in this section. Since the methods which we study in this thesis were derived for systems being a pure time-delay, an interesting question is: Is the method (fischer8) only suitable for pure time-delay systems or can it be used also for systems with dynamics? We will investigate this. We will only show results for low SNR. Results for high SNR are similar and can be found in [Bjö03e]. Outliers are removed by procedure 1 in Section 7.5.1. The number of trials was 4096.

The ANOVA table in [Bjö03e] says that there are differences between the systems. Figure 8.8 shows confidence intervals for pairwise comparisons of systems. We cannot say that there are any differences between the systems G_1, G_5 and G_6 but we can say that these three systems give significantly better estimation results than the other ones. The system G_2, which is faster, is not that good. The system G_9, i.e. the pure time-delay has the worst estimation performance. It is significantly worse than the other systems. This is strange because the used Laguerre domain methods were derived for pure-time-delay systems like G_9(Section 4.3). The validation graphs for the ANOVA and the confidence interval plot in Appendix A.3.2 are acceptable .

Figure 8.7 Confidence intervals (95% simultaneous confidence level) comparing Laguerre domain approximation methods with step input at SNR = 1. Positive transformation (RMS error)$^{0.96}$=> "The lower the better". Tukey's honestly significant difference criterion [Mat01] is used. (t224b1)



8.3.4 Execution time

The execution time of the method fischer8 is long. Table 8.1 shows the execution time for the methods fischer8 and arxstruc3 (Section 7.2) in a simple test in Matlab. The execution speed of fischer8 can probably be optimized. The most execution time is required for the calculation of the regression matrix Φ (Equations (4.39) and (4.42)). In our test the calculation of Φ took 82% of the total execution time ($\approx$the first column in Table 8.1). The execution time for Φ increases rapidly with increasing number $N+1$ of used Laguerre function. Fortunately, Φ depends only on the input signal and not on the output signal (see Equation (4.39)). This means that if we repeatedly use the same input signal, Φ needs to be computed only once. In the second column in Table 8.1 the already computed Φ is used. Some of the differences in execution time between the first and the second column in Table 8.1 is because Matlab does not load a function into memory before it is needed. Assumably, the whole difference for arxstruc3 between the first and later calls is due to this loading of functions into memory. The loading has a very small effect on the execution time for fischer8. Comparison of the execution time of more methods are done in Section 9.4.

Figure 8.8 Confidence intervals (95% simultaneous confidence level) for different systems for fischer8 with input signal Steps at SNR = 1. Positive Transformation (RMS error)$^{4.48}$=> "The lower the better". Tukey's honestly significant difference criterion [Mat01] is used. (t225b1.m)

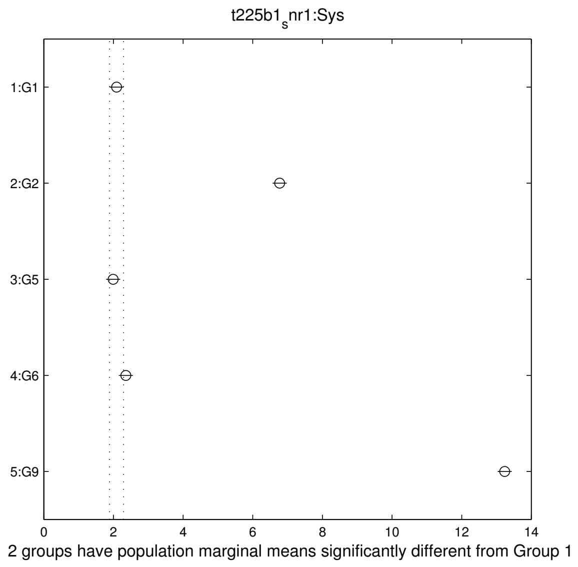


Table 8.1 Execution time for fischer8 and arxstruc3 implemented in MATLAB in seconds on a SUN SunBlade 100 computer. The column "Later calls" shows the mean execution time in calls 2 to 5. (t227b1)

Method	First call	Later calls
fischer8	63.994	2.882
arxstruc3	0.202	0.118

8.3.5 Discussion

Even if we only have studied the method fischer8 in Section 8.3.3, it is reasonable to believe that the same results should apply also for the other Laguerre domain methods (Section 7.2), i.e. a pure time-delay system is not the easiest system to estimate the time-delay for.

The Laguerre domain approximation methods are different from the method Laguerre DAP in Section 4.2.5 and in [IHD01b, Hor00]. In Laguerre DAP the impulse response of the system is modeled by Laguerre functions. In Laguerre domain approximation methods the input and output signals are modeled by Laguerre functions. Whether the modeling of a signal with Laguerre functions is successful depends on how similar the signal and the Laguerre functions are. Theoretically as $N \to \infty$ all finite length signals can be modeled exactly by Laguerre functions (independent of choice of the pole). But this is less interesting since we cannot use infinite sums in the reality. In order to represent a discrete-time signal with a finite length M in the frequency domain it is enough with M DFT (Discrete Fourier Transform) coefficients [GLM01]. For the discrete-time Laguerre domain this is probably not enough. A discrete-time signal with a finite length M is seen as signal of infinite length where only M signal values are different from zero. (See *extended spaces* in [Fis99, FM99c].) This means that we probably need many more Laguerre coefficients than M to exactly represent the signal. If, however, the truncated Laguerre functions are linearly independent, it should be enough with M Laguerre coefficients. Then Equation (4.34) will be a nonsingular quadratic linear equation system.

Modeling of impulse responses with Laguerre functions [Bjö02, IHD01b, Hor00] probably is more suitable than modeling input and output signals because typical impulse responses are more "Laguerre-like" than typical input and output signals. Compare Figure 4.11 with Figures 7.1 and 7.2-7.4.

Laguerre modeling of input and output signals can work well when the signals have a short extent in time and do not oscillate too much (e.g. are low-pass). See Figure 4.12. The pole α must be suitably chosen [Fis99, FM99c, FM99b, Bjö03e] (see also [Bjö03e]) and the number N of Laguerre functions must be large enough (Figure 4.12). Of our signals the signal with steps work fine but the other signals not so well.

Laguerre functions are chosen in [Fis99, FM99a, FM99b, FM99c] as the basis functions probably because the signals in the intended applications are easy to represent in the Laguerre domain. Other basis functions could also be possible, e.g. Kautz functions.

Laguerre coefficients with low subscripts are suitable for low-pass signals. For other signals perhaps other ranges of subscripts should be used.

8.3.6 Conclusions

We draw the following conclusions from the work about Laguerre domain methods in this thesis and in [Bjö03e].

- The input signal:
  - The input signal has a large influence on the time-delay estimation quality (Figure 8.6.
  - The Laguerre domain methods are suitable for input signals that can be well approximated by truncated series of Laguerre functions. The signals should be low-pass and not too long in time.
  - One or several steps (Figure 4.12) or a triangle windowed low-frequency sinusoid [Bjö03e] are examples of suitable signals. Random binary signals (Figure 4.12) are not suitable.

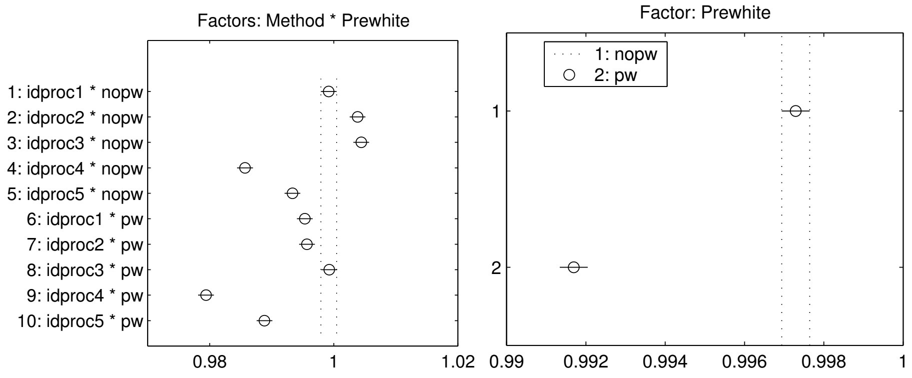
- The system:
  - The system has a smaller influence on the time-delay estimation quality than the input signal.
  - Systems with a not too fast dynamics give better time-delay estimation quality than pure time-delay systems despite the fact that the estimation methods were derived for pure time-delay systems (Figure 8.8).
- The estimation methods:
  - The method fischer8 (Section 7.2) is the best of the tested methods (Figure 8.7) because it gives the lowest RMS error on average.
  - The number of used Laguerre functions should be as high as possible (Figure 4.12). However, a high number gives a longer execution time. The choice will therefore be a trade-off between estimation quality and execution speed.
  - The execution time of the methods is high, especially if many Laguerre functions are used. The execution time can, however, be reduced for subsequent estimations if the same input signal is employed. See section 8.3.4 and Table 8.1.
  - The methods do not seem to be robust. They sometimes fail and deliver unreasonable estimates, e.g. negative or very large positive.

8.4 Continuous-Time One-Step Explicit Methods (Idproc Methods)

In this section the continuous-time one-step explicit methods (Section 5.1.1) idproc1-idproc5 (Section 7.2) with and without prewhitening are compared and their properties are studied in the standard benchmark (Section 7.3.1) using 128 trials. The estimated time-delay was not allowed to be larger than 30 (procedure 3 in Section 7.5.1). A negative transformation $(\text{RMS error})^{-0.011}$ was used (Figure A.15). This means that "the higher value the better" in the confidence interval plots (see Section 7.5.3). The ANOVA table, Table A.6 in Appendix A.4 tells us that all main factors and all factor interactions, for which we have plotted confidence intervals, have a statistical effect. The analysis result is depicted in Figure 8.9-8.10. The validation graphs in Appendix A.4 show that the prerequisites are not completely fulfilled but for the conclusions we draw below the results are clear (well separated confidence intervals). In Figure 8.9 it can be seen:

- It is best without prewhitening (nopw) of the input data (Figure 8.9).
- For nopw we cannot say that there is any difference between the model structures idproc1,2,3,5 on average but idproc4 is worst on average (Figure 8.9). Recall that idproc4 is the only model structure with three real poles. It has no complex poles.

Figure 8.9 Confidence intervals (95% simultaneous confidence level) in idproc methods for combinations of model structure and prewhitening (= method) to the left and only prewhitening to the right. The standard benchmark (Section 7.3.1. Negative transformation: (RMS error)$^{-0.011}$=> "The higher the better". (t233b2)



- Two of the best choices of method are idproc1*nopw and idproc2*nopw. We called these choices idproc6 and idproc7, respectively, in Section 7.2.

Not only the model structure matters. It is also important not end in a local minimum. Therefore, the initialization of the time-delay in the optimization is important. Compare with Sections 5.1.1-5.1.3.

In Figure 8.10 we see that:

- On average these methods work better for random binary input signals than step input signals.
- On average these methods seem to work better for slow (G_1) than fast (G_2) systems. This can be uncertain because, even if the intervals for G_1 and G_2 do not overlap, they are not so much separated and their length are uncertain because the prerequisites for the analysis was not completely fulfilled.

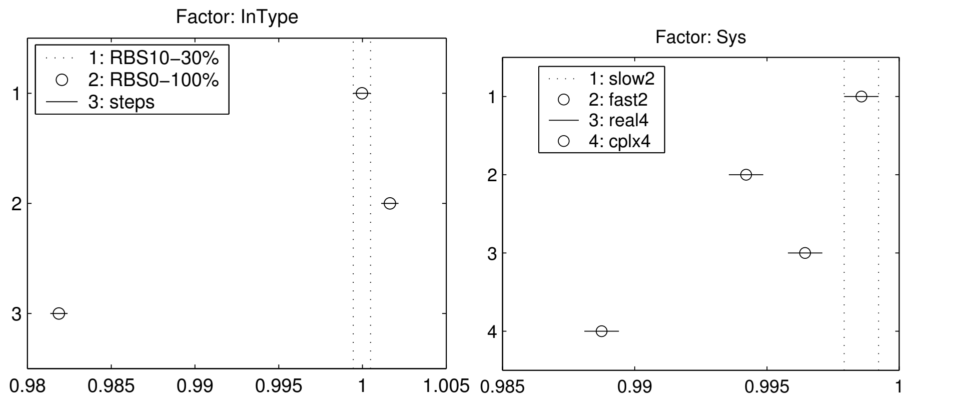
8.5 Discrete-Time One-Step Explicit Methods

In this section, parameters are chosen and properties are studied in some discrete-time one-step explicit methods (Section 5.2). These methods are here also called *arxstruc type methods.*

8.5.1 Choice of parameters in arxstruc type methods

The arxstruc type methods (Section 5.2) arxstruc, oestruc and met1struc (Section 7.2) were simulated in the standard benchmark (Section 7.3.1). The number

Figure 8.10 Confidence intervals for pair-wise comparisons (95% simultaneous confidence level) of input signal types (left) and systems (right) for the average over idproc1-idproc5 in the standard benchmark (Section 7.3.1. Negative transformation: (RMS error)$^{-0.011}$=> "The higher the better". (t233b2)



of trials was 2048 for arxstruc, 192 for oestruc and 512 for met1struc. The reason for the different number of trials is the very different execution times for the methods, cf.Section 9.4. Three method factors were varied during the simulations, namely the model orders n_a (or n_f) and n_b and using prewhitening (Section 7.3) or not.

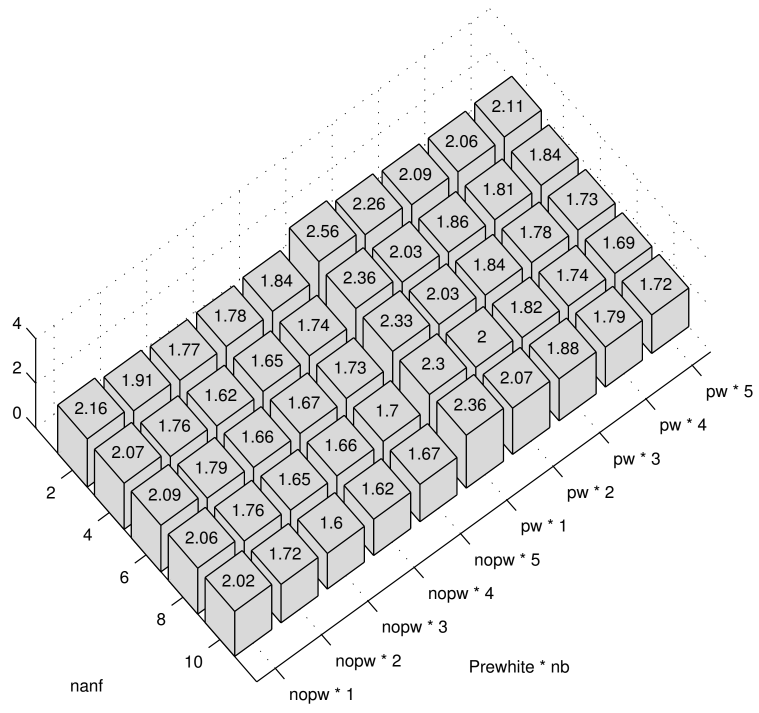
For the method arxstruc we see in Figure 8.11 that there are many values of n_a, n_b and prewhitening that give approximately the same average RMS error. The lowest RMS error is obtained for $n_a = 10$, $n_b = 3$ and without prewhitening in this simulation. However, we prefer $n_a = 10$, $n_b = 5$ and without prewhitening because these values gave the best result in a similar simulation. We have called this combination arxstruc3 in Section 7.2. A choice of parameters by the aid of confidence intervals was not possibly because the prerequisites were not fulfilled [Bjö03f].

For the method oestruc it is seen in Figure 8.12 that on average the best model orders are the lowest ($n_f = 2$ and $n_b = 1$) of the tested. If $n_b > 1$ this would enable more models to give a low optimization criterion value but with different time-delays. It is also seen in the same figure that without prewhitening is the best. In [Bjö03f] confidence intervals confirm that $n_f = 2$, $n_b = 1$ and no prewhitening is the best choice. We called this combination oestruc3 in Section 7.2.

In Figure 8.13 it is seen that on average the best model parameters for the method met1struc with are $n_a = 10$, $n_b = 1$ and without prewhitening. We called this combination met1struc3 in Section 7.2. A choice of parameters by the aid of confidence intervals was not possible because the prerequisites were not fulfilled [Bjö03f].

From Figures 8.11-8.13 and execution time measurements we see that

Figure 8.11 RMS error for arxstruc as a function of the model orders n_a and n_b and prewhitening or not. Axis labels and tick mark labels are explained in Section 7.5.2. (t234b2)

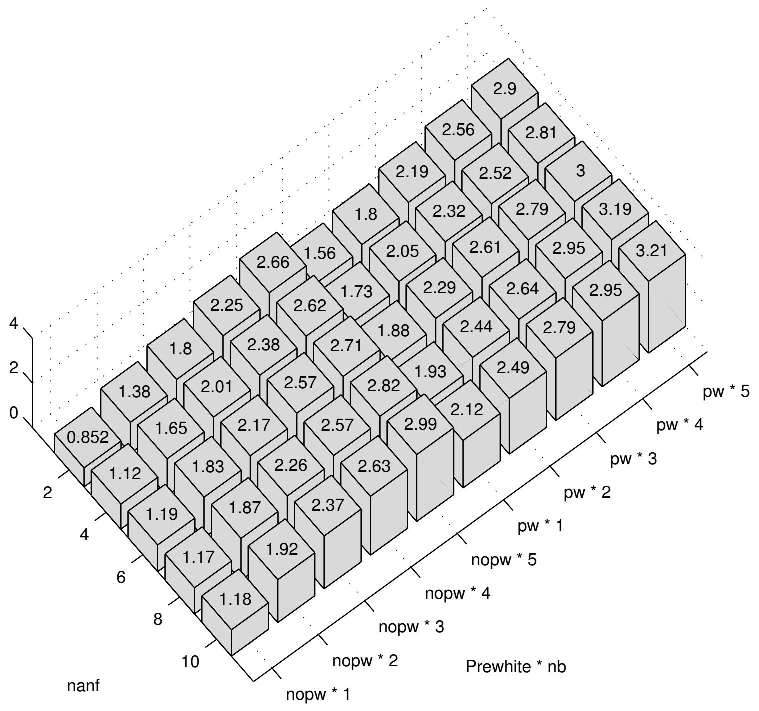


- oestruc has lower RMS error (0.85 sampling intervals) than arxstruc and met1struc but is very slow (One estimation took 13.0 s on a SunBlade 100 computer).
- met1struc has a RMS error that is nearly as good as for oestruc (met1struc: 0.95 sampling intervals) and is fast (One estimation took 0.741 s).
- arxstruc has a higher RMS error (1.6 sampling intervals) but is very fast (One estimation took 0.143 s).

8.5.2 More properties of arxstruc type methods

In this section another simulation is used. The three methods arxstruc3, oestruc3 and met1struc3 (Section 7.2) were put into service for high SNR (=100) in the standard benchmark (Section 7.3.1). The worst estimates were removed by procedure 1 in Section 7.5.1. The transformation (Section 7.5.3) became negative: (RMS error)$^{-0.12}$. This means the higher value in the confidence interval plot the better. We see in Figure 8.14 that

Figure 8.12 RMS error for oestruc as a function of the model orders n_f and $n_b$and prewhitening or not. Axis labels and tick mark labels are explained in Section 7.5.2. (t236b2)



- On average these methods work better for fast than slow systems.
- On average these methods work better for random binary input signals than step input signals.

8.5.3 Discussion

We found in Section 8.5.1 that oestruc is the best method in the tested cases. Also Swanda [Swa99] consider that oestruc is better than arxstruc. It is not surprising that oestruc is better than arxstruc since the tested systems have OE structure. This also helps met1struc to give good results.

When estimating a discrete-time state space model (zoh sampling) of a system with a long time-delay (longer than the sampling interval) the order of the model will increase with one for each sampling interval of the time-delay [ÅW84, p. 42]. This could indicate that the used order 10 of the state space model in met1struc could be too low for long time-delays. If the continuous-time time-delay is 9 sampling intervals, a model order of 10 seems to be on the limit to be too low. Another

Figure 8.13 RMS error for met1struc as a function of the model orders n_a and n_b and prewhitening or not. Axis labels and tick mark labels are explained in Section 7.5.2. (t235b2)

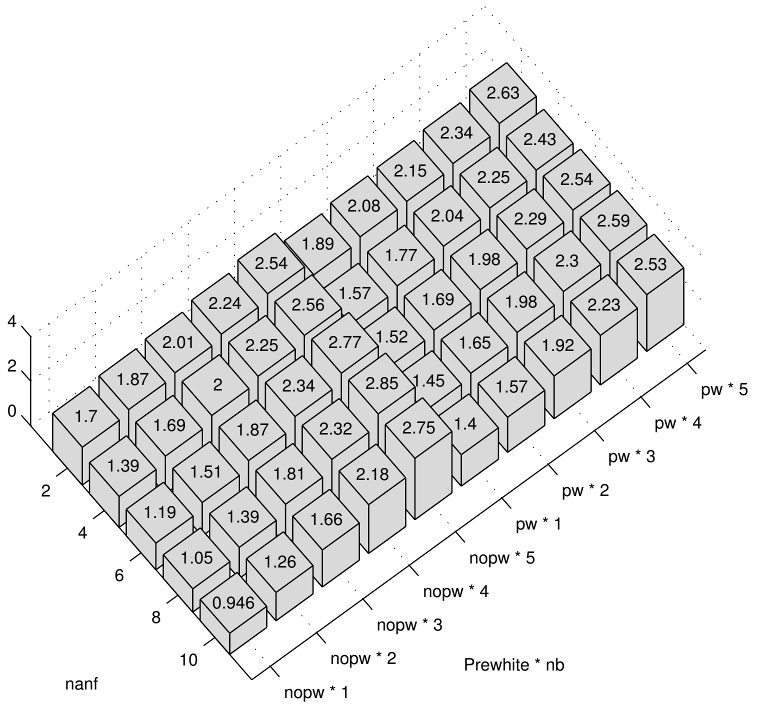
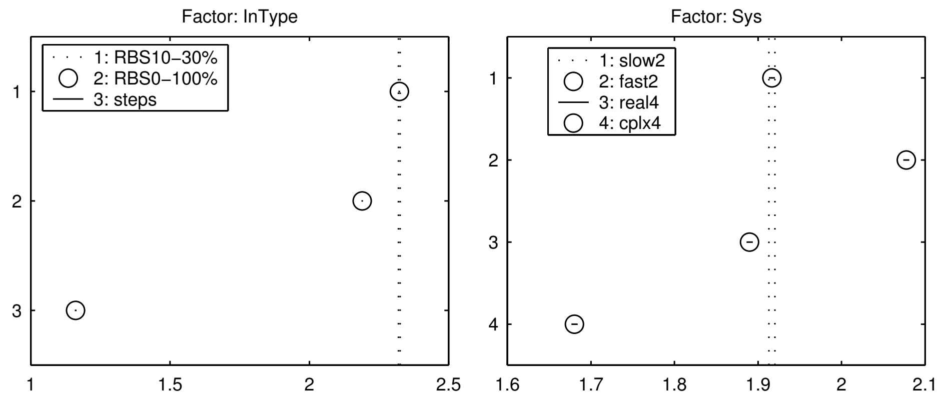


Figure 8.14 Confidence intervals for pair-wise comparisons (95% simultaneous confidence level) of input signal types (left) and systems (right) for the average over arxstruc3, met1struc3 and oestruc3. SNR = 100. Negative transformation: $(\text{RMS error})^{-0.12}$=> "The higher the better". (t208b15)



way for the state space model to handle the time-delay is to to approximate it with non-minimum phase zero(s). In this way a lower model order can be sufficient. This is also what happens in met1struc. However, for longer time-delays than used in this thesis, it would be advisable to use a higher fixed model order or to chose the model order automatically to give a good model. This can be done by giving 'best' as the input parameter order to the function n4sid (Algorithm 9) in the Matlab System Identification Toolbox.

The advantage of met1struc over oestruc is the lower execution time. A disadvantage is that it is more complicated. In applications where the time-delay is changing but the noise does not change it should be possible to estimate the noise model once off-line and use it in many subsequent time-delay estimations with a modified met1struc method. It is not necessary to estimate this noise model with a subspace state space method as in Section 5.2.3 but can by done by a less complicated method.

In Section 8.5.1 the best choice of model orders for arxstruc was the highest of the tested n_a and n_b. The reason for this is probably that high orders are needed to approximate the noise system well by $1/A$ since the true systems are not of ARX structure. An all-pole system $1/A$ of enough high order should be able to approximate the noise system enough well. Such an approximation is used in [FMS91, p. 655]. See also the discussion in Section 5.2.3 about bias in the model G for different cases of system/model structures and orders.

In Section 8.5.1 the best choice of model orders for oestruc was the lowest of the tested ($n_f = 2$ and $n_b = 1$). These orders are enough to accurately model the true system. The true systems are either of second or fourth order. This would mean that $n_f = 2$ or $n_f = 4$ would be appropriate. On average $n_f = 2$ is apparently better. If the order n_b were higher than 1 it would introduce an ambiguity for the time-delay. For example, both the true time-delay and the true time-delay minus one (with the the first B-parameter equal to zero) would give good fits to the data. Therefore it is understandable that $n_b = 1$.

The theoretical explanation for the choice of model orders for met1struc in Section 8.5.1 is not clear.

8.6 Area and Moment Methods

In this section properties are discussed in area and moment methods (Sections 6.1-6.2). The objective of using area and moment methods was to by integration avoid the sensitivity of a single or a few values of uncertain step and impulse response estimates. Here, that among other things are discussed.

In [Bjö03b] some observations about area and moment methods are made. Here they are summarized:

- The area and moment methods often fail to deliver an estimate. The methods fail when somewhere during the computations unreasonable results appear, e.g. when an area that should be positive becomes negative. The reason is uncertain estimates of the step and impulse responses. This should be a much smaller problem with measured step and impulse responses as is suggested in [ÅH95].

- Moment methods seems to be more unreliable than area methods. Not even with a true simulated impulse response they give a good answer. For these methods the percentage of failed estimations was very high (>54% for all combinations of input signals and SNR in the standard benchmark in Section 7.3.1 but with a different definition of the SNR) in a Monte Carlo simulation [Bjö03b]. Therefore these methods are not useful.
- The uncertainty of the step response estimate often increases with the time. Since the gain K, which is used in the area methods, of the system is estimated by the last step response coefficients this estimate often will be completely incorrect. This has a very negative effect on the time-delay estimation. Fast systems seem to have a larger uncertainty in the step response than slow systems.
- For the area methods (Section 7.2) there are several things that can go wrong:
  - The estimated gain $\hat{K}$ can get negative if the estimated step response falls below zero at the end of the estimated step response.
  - The estimated area $\hat{A}_0$ can become negative if too much of the estimated step response is above the estimated gain $\hat{K}$. This can happen if the estimated gain is too low.
  - The time T_{ar} can be estimated to a too high value. This can happen if the gain K is estimated to a too low value.
  - The time-delay estimate can be calculated to a negative value. This happens if either or both $\hat{T}_{\text{ar}}$ is too low and $\hat{T}$ is too large. The time constant $\hat{T}$ may become too large if the estimated gain $\hat{K}$ is too low.
- For area and moment methods, the slow low order system (G_1) is easier than the the fast system (G_2). The reason is that the integrals in the area and moment methods can be better estimated when the function to integrate (impulse or step response) does not change rapidly. This is the inverse of the behavior of many other methods, e.g. thresholding of impulse response, which can easier detect a rapid change than a slow one.
- It is intuitive that fast system is harder for area methods because the A_1 integral (Equation 6.4) will be done over a very short time span where the step response is nonzero.

In [Bjö03b] method parameters were chosen. Below, the terms 1stord and 2ndord mean the first and second order model structures in Section 6.1-6.2. The terms trapez1 and sum1 are two ways to integrate numerically [Bjö03b]. The term nopw means without prewhitening the input signal.

- Area methods without prewhitening is better than with prewhitening.

- For the area methods there were two combinations of method parameters which were better than the other. One of them, the combination nopw*1stord*trapez1, we call area2.
- There is one combination of method parameters for moment methods which seems to be better than the other: nopw*1stord*sum1. This combination we call moment2.

Some interesting points are:

- The uncertainty of the estimates of the impulse and step response estimates are not used in the area and moment methods. If we did not throw away this information we probably could obtain better estimates of the time-delay.
- In the area methods we numerically integrate twice, first from impulse to step response, then to A_0 and A_1 (Equations 6.3 and 6.4). There will be numerical errors in both integrations. This could deteriorate the time-delay estimate. But still the area methods are better than the moment methods.
- It is clear that it is important to estimate the gain K (Equations 6.1 and 6.2) correctly in area methods.
- Moment methods are weighting up the later parts of the impulse or step response. This can give large errors because the signal to noise ratio (SNR) is lowered (compare with Section 6.3).

9

Comparing Time-Delay Estimation Methods in Open and Closed Loop

In the previous chapter, parameters of time-delay estimation (TDE) methods were chosen. In this chapter, those methods with the chosen parameters are compared using Monte Carlo simulations, mainly regarding the estimation quality but also regarding the execution time. In Section 9.1 results for open loop simulations with fixed systems are displayed, in Section 9.2 results for open loop with random systems and in 9.3 for closed loop with fixed systems. Execution times are shown in Section 9.4.

9.1 Open Loop Simulations with Fixed Systems

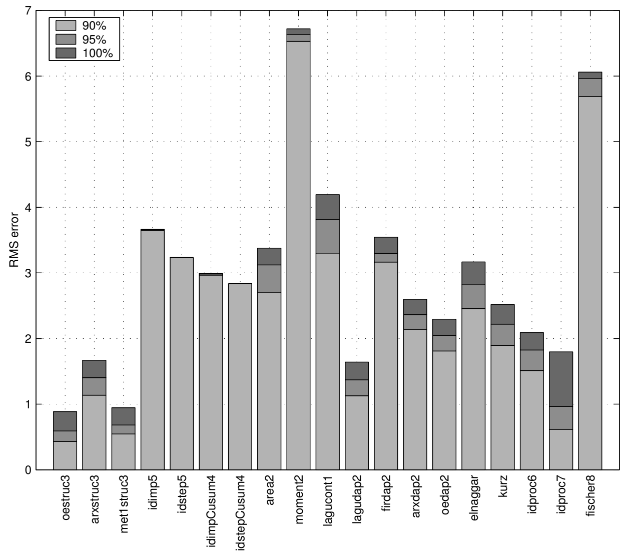
In this section several TDE methods are compared in open loop with fixed systems.

9.1.1 The standard benchmark

This section contains results from comparing several methods in open loop simulations with fixed systems (The standard benchmark in Section 7.3.1). The methods are defined in Section 7.2. See also the list of methods on page 2.

Figure 9.1 shows the average (over all factors) RMS estimation error for several methods when the 90%, 95% and 100% best estimates are retained (procedure 1 in Section 7.5.1). The methods oestruc3 and met1struc3 are the best methods when 100% are used. For 90%-95%, they are challenged by idproc7. The methods oestruc3 and met1struc3 are better than arxstruc3. The method idproc7 is better than idproc6. The method idproc7 mostly gives very accurate estimates but sometimes fails. It is favored by only retaining the 90%-95% best estimates. The method

Figure 9.1 Average RMS error for different methods in open loop with fixed systems. Average over the environment factors in the standard benchmark (Section 7.3.1). (t208b8)

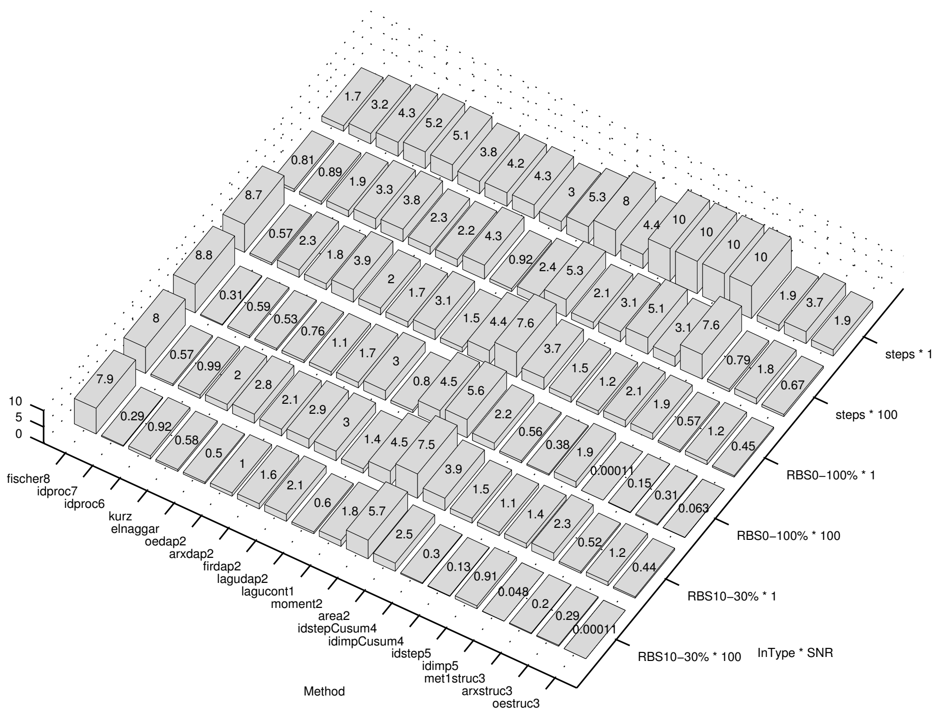


lagudap2 also belongs to the best methods. The methods lagucont1, idimp5, idstep5, idimpCusum4, idstepCusum4, area2, moment2 and fischer8 often fail.

Figures 9.2-9.3 show the RMS estimation error as a function of the factor levels. Figure 9.2 displays the RMS error for different combinations (*cases*) of input signal and SNR when the 95% best estimates are used. Here we can see which method to choose when the input signal and/or the SNR is given. We can also choose the best input signal. We see that RBS signals are better than step signals. For high SNR there are more methods among the best than for low SNR. The discrete-time one-step methods oestruc3 and met1struc3 belong to the best methods in all cases. The time domain approximation methods are very inaccurate for step inputs but can be really accurate for RBS and high SNR. The area and moment methods and lagucont1 are not good in any case. The other frequency domain approximation methods (lagudap2, firdap2, arxdap2 and oedap2) and elnaggar are neither among the best nor the worst methods for any case except for lagudap2, which is among the best for steps with high SNR. The method idproc7 is among the best methods for steps and with low SNR for RBS signals. For high SNR and RBS the method idproc7 is beaten by several methods. The method idproc6 is not as good as idproc7 for any case. The method fischer8 is very inaccurate for RBS signals but among the best for step input signals. It is especially good for low SNR.

Figure 9.3 displays the RMS error for different combinations of method and system. The reason why the first order model structure idproc1 gives good results

Figure 9.2 Average RMS error for different methods, SNR and input signal types in open loop with fixed systems. The 95% best estimates are retained (Section 7.5.1). Axis labels and tick mark labels are explained in Section 7.5.2. (t208b9)



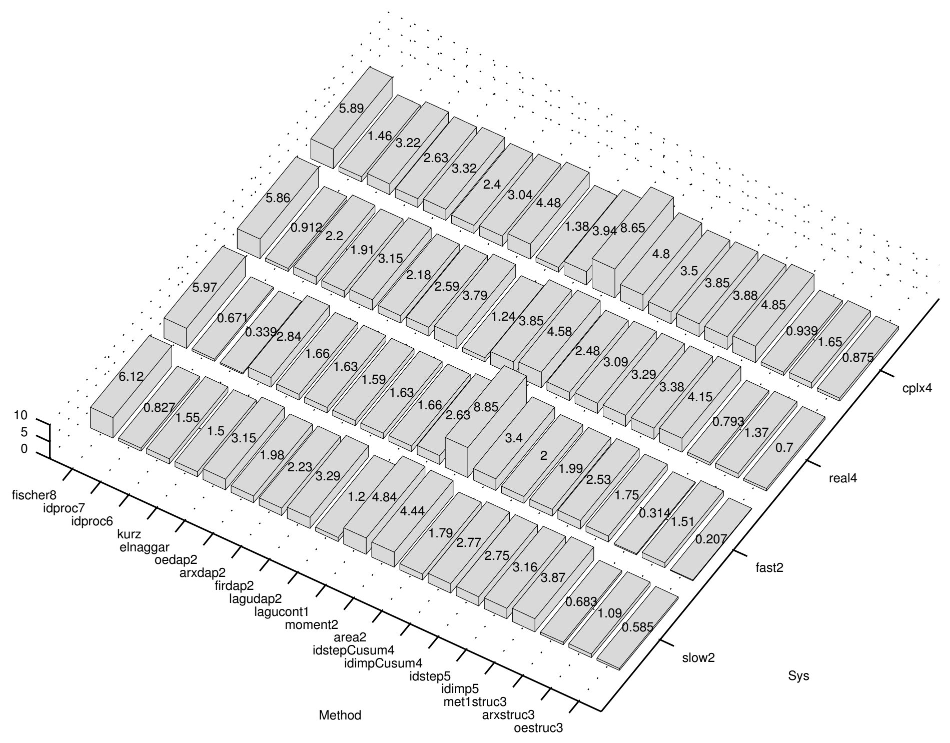
with the second order system G_2 is probably because one of the time constants of G_2 is very fast (Equation 7.3) and therefore G_2 can be well approximated with a first order system.

All the methods in Figures 9.1-9.3 could not be analyzed together by ANOVA and confidence intervals because the methods behave very differently, which makes it difficult to fulfill the requirements (Section 7.5) of this type of analysis.

9.1.2 Confidence intervals for step input

This section compares some promising methods (oestruc3, met1struc3, lagudap2, idproc6, idproc7 and fischer8) for the case when we cannot choose other input signals than steps. The analysis is performed with ANOVA and confidence intervals for pair-wise comparisons when the SNR was low (SNR=1). Table A.8 in Appendix A.6 shows the ANOVA table and Figure 9.4 confidence intervals for pair-wise comparisons of the methods. Appendix A.6 also shows validation graphs of

Figure 9.3 Average RMS error for different methods and systems in open loop with fixed systems. Average over the environment factors in the standard benchmark (Section 7.3.1). The 95% best estimates are retained (Section 7.5.1). Axis labels and tick mark labels are explained in Section 7.5.2. (t208b9)

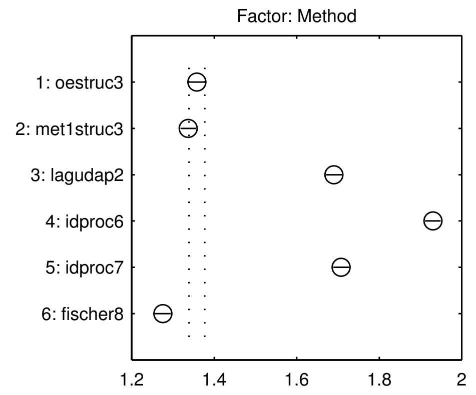


the prerequisites for the ANOVA and confidence intervals. For the interpretation of the table and graphs, see Section 7.5.3.

When we look at Figure 9.4, we cannot say that there is any significant difference between oestruc3, met1struc3 and fischer8 (overlapping or nearly overlapping intervals) but we can say that these methods are significant better than the other methods in the graph. We also see that the difference between lagudap2 and idproc7 is not significant but these two methods are significantly different from the other methods in the graph. The method idproc6 is significantly worse than the the other methods in the graph. The standard deviation of the residuals (Figure A.23) is not constant, but the above result is very clear so that the non-constant standard deviation does not change our conclusions. The residuals are near Gaussian as desired (Figure A.22).

For the high SNR (SNR=100) an analysis with ANOVA and confidence intervals was not possible because the prerequisites for the analysis was not fulfilled.

Figure 9.4 Confidence intervals (95% simultaneous confidence level) for some promising methods (Section 9.1.2) with input signal Steps at SNR = 1 in open loop with fixed systems. Positive transformation (RMS error)$^{0.48}$=> "The lower the better". Tukey's honestly significant difference criterion [Mat01] is used. See Section 7.5.3 for the interpretation of the graph. (t208b10.m)



9.2 Open Loop Simulations with Random Systems

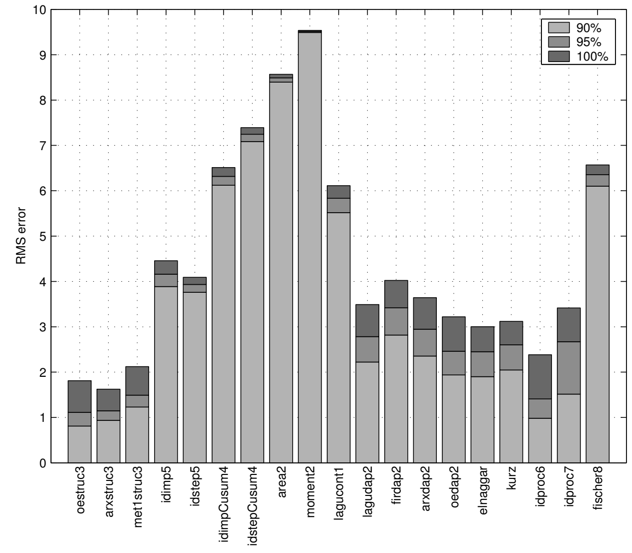
This section contains results from comparing several methods in open loop simulations with random systems. A short description of the methods can be found in Section 7.2.

The continuous-time random systems are of 10th order, can have both positive and negative zeros but only negative poles (stable). The systems are integrating (pole at zero) or non-integrating and have positive or negative random gain. The time-delay is random between 0 and 15 sampling intervals.

Figure 9.5 shows the average (over all factors) RMS estimation error for different methods when the 90%, 95% or 100% best estimates are kept (procedure 1 in Section 7.5.1). The results are similar to the ones for fixed systems (Figure 9.1). The methods idproc6, idproc7 and oestruc3 give astonishing good results for the 10th order systems despite the fact that they use low order models. They are treated unfairly compared to methods using high order models, e.g. arxstruc3, met1struc3.

The RMS error is for most methods somewhat higher than for the fixed systems used in Figure 9.5. The RMS error of arxstruc3, firdap2, elnaggar and idproc6 seems to be nearly unchanged. The CUSUM and area methods have a much higher RMS error, probably because these implementations are not adapted to all types of systems. Also lagudap2 has a much higher RMS error and is on approximately the same level as firdap2, arxdap2 and oedap2.

Figure 9.5 Average RMS error for different methods on random systems in open loop. (t215b2)



9.3 Comparing the time-delay estimation methods in closed loop

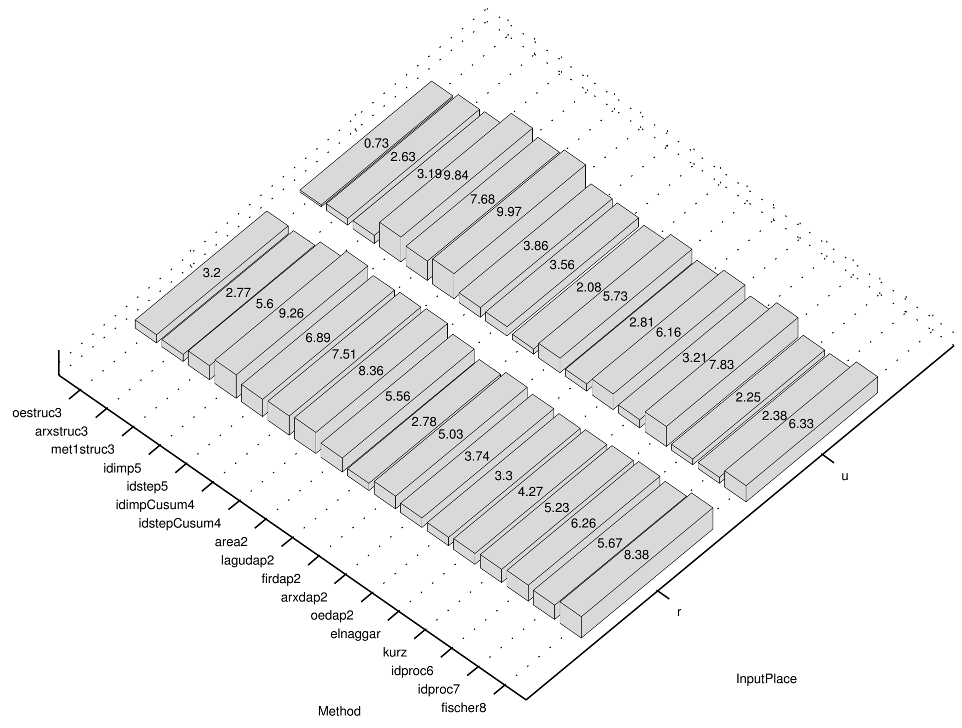
9.3.1 Closed loop simulation results

This section contains results from comparing several methods in closed loop simulations with fixed systems. The simulation setup is described in Section 7.4. The same methods as in Sections 9.1-9.2, but without moment2 were used. They are described in Section 7.2. We have plotted the RMS error of the time-delay estimates. The worst estimated values were removed by procedure 1 in Section 7.5.1 (95% kept).

In Figure 9.6 the method oestruc3 using the system input $u(t)$ seems to be the best method. This is in accordance with the results in [Swa99]. Also in open loop (Sections 9.1-9.2) oestruc3 was the best method. Most methods are better with the system input $u(t)$ than the reference signal $r(t)$. There are several methods that compete for the second best result, e.g. arxstruc3, lagudap2, arxdap2, idproc6 and idproc7, all using $u(t)$. If subsample time-delay estimates are needed, the methods lagudap2, idproc6 and idproc7 seems to be the best.

For the interested reader, Figures 9.7-9.9 show the RMS estimation error as a function of the system, controller and noise model, respectively. Since the same simulation setup was used in [IHD00, IHD01a, IHD01b] comparisons are possible.

Figure 9.6 Average RMS error for different methods in closed loop. The 95% best estimates are retained (Section 7.5.1). Axis labels and tick mark labels are explained in Section 7.5.2. (t176d2)



9.3.2 Discussion

According to [Lju99, p. 433], when the noise model is fixed $H(q,\theta) = H_*(q)$, the estimated model will in closed loop converge to

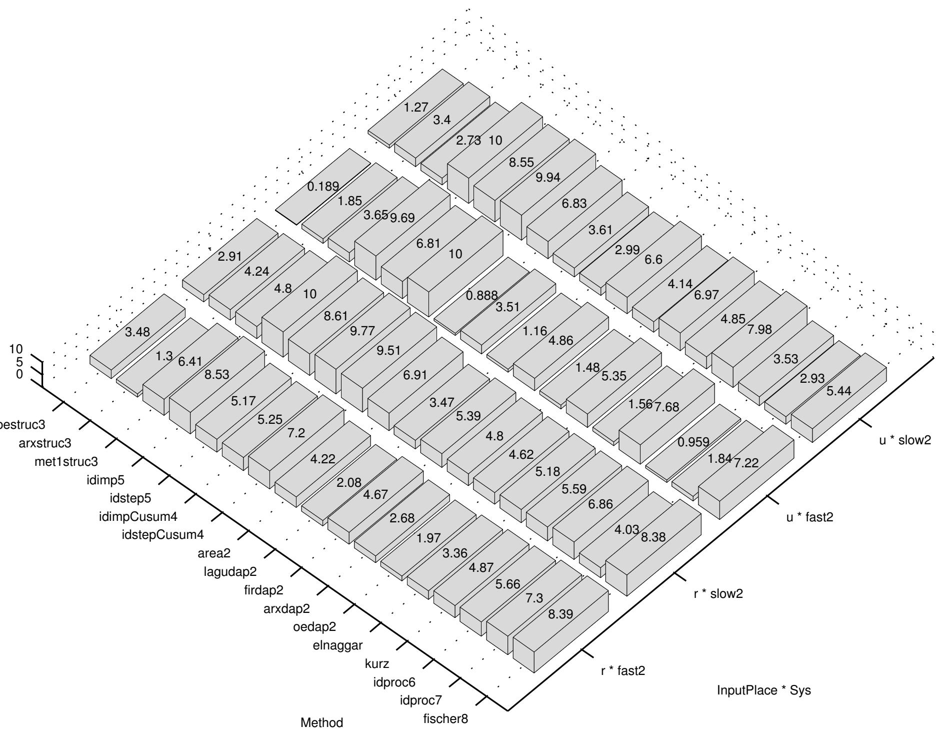
$$G_* = \arg\min_\theta \int_{-\pi}^{\pi} |G_0(e^{i\omega}) + B(e^{i\omega}) - G(e^{i\omega},\theta)|^2 \cdot \frac{\Phi_u(\omega)}{|H_*(e^{i\omega})|^2} d\omega,$$

when the number of data $N \to \infty$, where the bias term $B(e^{i\omega})$ is

$$B(e^{i\omega}) = \frac{\lambda_0}{\Phi_u(\omega)} \cdot \frac{\Phi_u^e(\omega)}{\Phi_u(\omega)} \cdot |H_0(e^{i\omega}) - H_*(e^{i\omega})|^2.$$

This means that when estimating OE models in closed loop there will be bias in the estimate if the noise model $H_*(q)$ is not equal to the true noise system $H_0(q)$, even if the system model $G(q,\theta)$ is capable of describing the true system $G_0(q)$. In open-loop, on the other side, there will be no bias for an OE model, irrespective

Figure 9.7 Average RMS error for different methods and systems in closed loop. The 95% best estimates are retained (Section 7.5.1). Axis labels and tick mark labels are explained in Section 7.5.2. (t176d2)

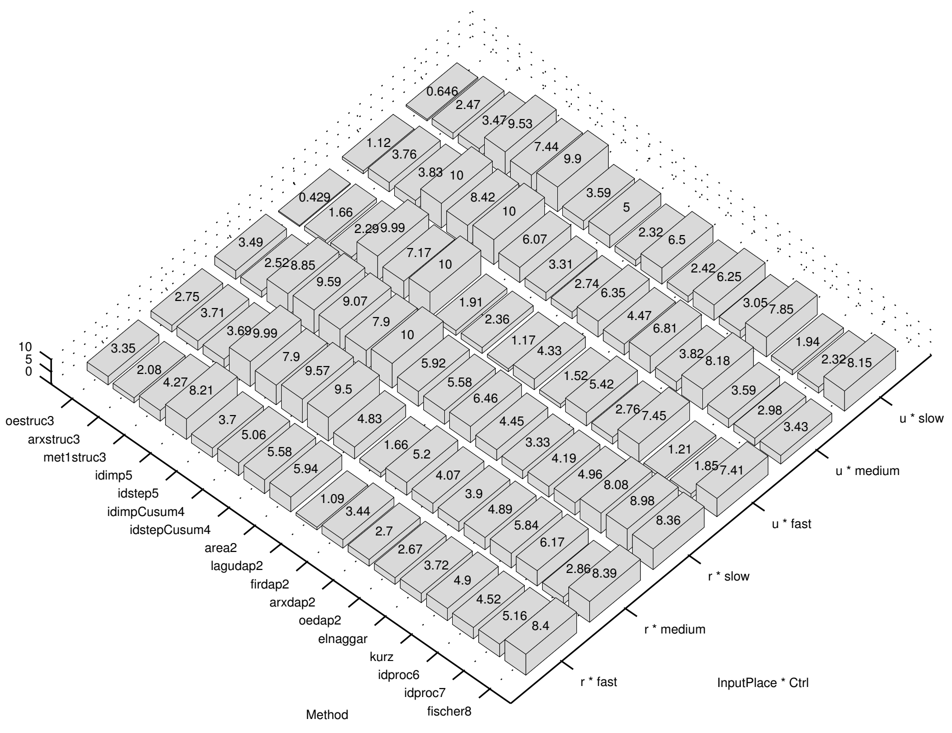


of the true noise system, if the system model $G(q, \theta)$ is capable of describing the true system $G_0(q)$ [Lju99]. In Figure 9.9 for the method oestruc3 using the system input u the RMS error seems to be better when the noise is white noise (oe). This is in accordance with the theory. When using the reference signal r white noise does not seem to be easier (against the theory) but using r is less interesting since it gives a higher RMS error.

9.4 Execution Time

Table 9.1 show the execution time on a SUN SunBlade 100 computer of the methods in this thesis. The method elnaggar is very fast. One call took 0.017 seconds. Second fastest is arxstruc3. Third fastest are firdap2 and arxdap2. After that there is a large group consisting of met1struc3, idimp5, idstep5 idimpCusum4, idstepCusum4, area2, moment2, lagucont1, lagudap2, oedap2 and kurz with an execution time around one second. Then comes fischer8 for subsequent calls if the same input

Figure 9.8 Average RMS error for different methods and controllers in closed loop. Axis labels and tick mark labels are explained in Section 7.5.2. (t176d2)



signal is used. The methods oestruc3, idproc6 and idproc7 are the slowest methods if the first call to fischer8 is excluded. They have an execution time of nearly 10 seconds.

If different input signals are used in all calls, the absolutely slowest method is fischer8 with 65.5 seconds . If using the same input signal in all calls, the execution time for the first and subsequent calls to fischer8 are very different. The first call to fischer8 took 65.15 seconds (as above) but subsequent calls took 2.86 on average. The reason is explained in Section 8.3.4.

Figure 9.9 Average RMS error for different methods and noise models in closed loop. The 95% best estimates are retained (Section 7.5.1). Axis labels and tick mark labels are explained in Section 7.5.2. (t176d2)

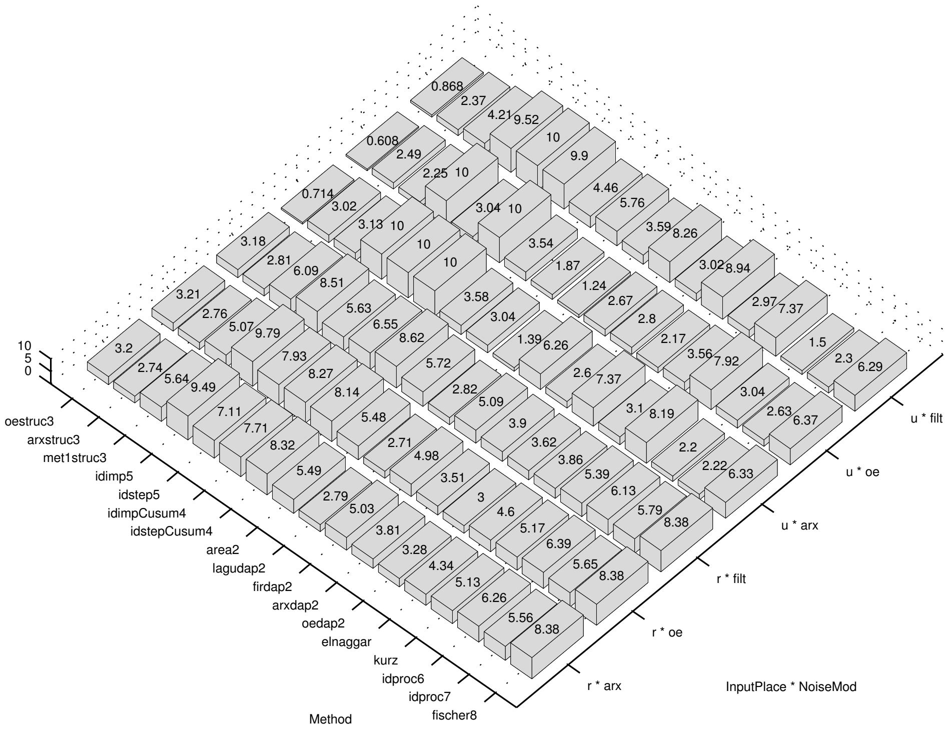


Table 9.1 Execution time for many methods in seconds on a SUN SunBlade 100 computer. (t227b1)

Method	Time	Method	Time	Method	Time
oestruc3	9.77	area2	1.26	elnaggar	0.017
arxstruc3	0.12	moment2	0.97	kurz	1.07
met1struc3	0.82	lagucont1	0.81	idproc6	8.40
idimp5	0.69	lagudap2	0.83	idproc7	9.76
idstep5	0.74	firdap2	0.22	fischer8 (first call)	65.5
idimpCusum4	0.68	arxdap2	0.22	fischer8 (subsequent calls)	2.86
idstepCusum4	0.74	oedap2	0.70		

10

Discussion and Conclusions

This chapter contains further discussion in Section 10.1. Then, in Section 10.2, recommendations for the choice of time-delay estimation (TDE) method for different cases are given. Following, Section 10.3 contains conclusions about properties of different classes (Chapter 3) of TDE methods. Finally, in Section 10.4 some suggestions for future work are listed.

10.1 Additional Discussion about Open Loop

In this section we will give some additional discussion about open-loop. The TDE methods are described in Section 7.2.

The methods idimp5, idstep5, idimpCusum4 and idstepCusum4 often miss to detect the start of the impulse response (Sections 4.1.1 and 4.1.3), especially for low SNR, because of noisy and uncertain impulse and step response estimates [Bjö03d]. Prewhitening of step input does not make idimp4 perform better (Section 8.1.1). Prewhitening can make the cross-correlation contain fewer maxima that compete with the correct one [Car93]. Maybe prewhitening amplifies the noise in certain frequency regions and thereby counteracts the increase in estimation performance.

The method lagudap2 performs well in most cases (but not best) and seldom really bad. Two reasons for lagudap2 being better than firdap2, arxdap2 and oedap2 are probably that typical impulse responses can be described well by Laguerre functions and that the model orders of the latter methods are not optimal. The model orders of the DAP methods should also have been selected via Monte Carlo simulations, which was not done in this thesis (except for the zero guarding). The results in [Hor00] for lagudap2 is in agreement with our results. lagucont1 often fails, probably because it has no protection against noise-corrupted zeros. The results in [Hor00] for lagucont1 is better than our results. Perhaps, the implementation in [Hor00] has some protection against bad zeros.

idproc7 is better than idproc6 as it has a more suitable model structure for the used systems. Both these model structures have a too low order for some of the tested systems. Therefore they will have some bias. Since oestruc3 and met1struc3 have the correct model structure (output error) they are better than arxstruc3.

area2 and moment2 often give very inaccurate estimates due to poor estimates of step and impulse responses [Bjö03b, Ing03]. These methods would probably perform better with measured step and impulse responses as in [ÅH95]. Another improvement is described in [Ing03].

The method fischer8 often give very poor estimates. This is probably due to its inability to describe certain signals in the Laguerre domain [Bjö03e]. On the other hand, fischer8 performs astonishingly well for steps, especially at low SNR. Perhaps it would be even better with more Laguerre coefficients. The used number, 150, was on the limit of being too computer intensive.

Some of the tested methods can estimate subsample time-delays. In, for example, time-delay of arrival estimation of the direction of impinging electro-magnetic waves in signal intelligence [HR97, FHJ02] or radar this is necessary.

10.2 Recommendations for Choice of Estimation Method

This section gives recommendations on which TDE method to choose in different cases. The methods are described in Section 7.2.

In open loop the method oestruc3 is best on average with respect to estimation quality (Sections 9.1-9.2). If sub-sample time-delay estimates are needed, the best method (Section 9.1) is idproc7 but keep in mind that the model structure should be the same as the true system. On the negative side, both oestruc3 and idproc7 have long execution times, which could be a limiting factor in some applications. There are other, much faster, methods, e.g. met1struc3, arxstruc3 and lagudap2, with nearly as good estimation quality.

Some of the tested methods can deliver estimates of subsample time-delays. These are the frequency domain methods, e.g. lagudap2, and the continuous-time one-step explicit methods, e.g. idproc6 and idproc7.

Recommendations for the choice of input signal and estimation method in open loop for the best estimation quality are:

- If you cannot use other input signals than steps: For high SNR use oestruc3, met1struc3, lagudap2, idproc7 or fischer8. If the SNR is low use only oestruc3, met1struc3 or fischer8. If subsample time-delay estimates are needed for low SNR use lagudap2 or idproc7 (Figures 9.2 and 9.4).
- If you can choose the input signal, use RBS (Random Binary) signals. For high SNR the best methods are oestruc3 and idimp5. There are, however, more methods giving good results. If subsample time-delay estimates are

Figure 10.1 Classes of active time-delay estimation methods (Chapter 3).

- Time-delay estimation methods
  - Time domain approximation methods (= thresholding methods)
  - Frequency domain approximation methods (= phase methods)
  - Laguerre domain approximation methods
  - ...
- Explicit time-delay parameter methods
  - One-step explicit methods
    - Continuous-time one-step methods
    - Discrete-time one-step explicit methods
  - Two-step explicit methods
  - Sampling methods
- Area and moment methods
- Higher-order statistics methods

desired, use idproc7. For low SNR use oestruc3, met1struc3 or idproc7. If the SNR is unknown, use only oestruc3 (or idproc7 for sub-sample time-delays). See Figure 9.2.

Also in closed loop with step input the method oestruc3 is best on average with respect to estimation quality. If sub-sample time-delay estimates are needed, the best methods are idproc6, idproc7 and lagudap2. For these methods use the system input signal (not the reference signal) as the input to the estimation method.

Note, that the method lagudap2 uses zero guarding (Section 4.2.7), which is a modification to the original method described in [Hor00, IHD01b].

10.3 Conclusions about Parameters and Properties of Methods

We have made a classification of existing time-delay estimation methods according to underlying principles (Chapter 3 and Figure 10.1). Different classes have different properties and are suitable in different cases. Some methods are, however, clearly inferior to others.

Summary of comparison and properties of estimation methods:

- The best result in open loop is achieved with prediction error methods, e.g. oestruc3 and idproc7 (Section 7.2), where the time-delay is an explicit parameter (belong to one-step methods) (Sections 9.1-9.2). The continuous-time methods of this class can estimate subsample time-delays.
- Thresholding methods (belong to the time domain approximation methods) do not perform so well. The reason is problems to discover the start of the impulse response from fluctuating estimates.

Table 10.1 This table tells if some methods are significantly better for slow than fast systems (an X in the "slow" column), the opposite (an X in the "fast" column) or if nothing can be said (no X). The same applies for RBS and steps as input.

	System		Input	
Method	slow	fast	RBS	steps
Thresholding methods (Sections 4.1 and 8.1.2)		X	X	
DAP methods (Sections 4.2.5 and 8.2)		X	X	
Laguerre domain methods (Sections 4.3 and 8.3)	X			X
Idproc methods (Sections 5.1.1 and 8.4)			X	
Arxstruc type methods (Sections 5.2 and 8.5.2)		X	X	
Area and moment methods (Sec. 6.1-6.2 and 8.6)	X			

- DAP methods (belong to the frequency domain approximation methods) are among the better regarding the estimation quality and they can estimate subsample time-delays.
- Laguerre domain approximation methods give good estimation quality for certain types of input signals but very poor for others.
- Area and moment methods using estimated step and impulse responses are hardly useful.
- A summary of which system and input signal is "better" for different TDE method classes is given in Table 10.1.

Special conclusions about thresholding methods (Section 4.1) in open loop are:

- The PEM [Lju99], matched filter [Hän91], correlation analysis [Lju99], cross-correlation method [Car93] and maximum likelihood [JD93] for estimating the impulse response and the time-delay are in principle the same thing (Section 4.1.2 and 4.1.5).
- Because the impulse response estimates are very uncertain, it is difficult to find the beginning of the true impulse response. Better methods to find the beginning than a simple threshold are needed. There are two parts in this. First, the uncertainty of the impulse response estimate should be lowered. Second, the change time must be estimated by going backwards from the detection time. Integration to step response or CUSUM thresholding give some improvement (Figure 8.2) but it is not enough (Figure 9.1). Some more advanced approaches are described in [CHWW99, Isa97, KG81], one of which is tested in our simulations (kurz) [KG81].
- Prewhitening has no large influence on the estimation quality. On average these methods work better for fast than for slow systems and better for random binary input signals than for step input signals. See Figure 8.1 and [Bjö03b, Bjö03d].

Conclusions about frequency domain methods (Section 4.2) in open loop (Some more more detailed conclusions about DAP methods can be found Section 8.2.4.):

- Time-delay estimates relying on the allpass part of the system estimate can be very incorrect if zeros of the system fall on the incorrect side of the stability boundary due to the noise (Section 4.2.6). A solution is to remove zeros near the stability boundary on the incorrect side (Section 4.2.7).

Conclusions about the Laguerre domain methods which are tested in this thesis (Section 4.3) in open loop (More detailed conclusions can be found in Section 8.3.6.):

- These methods are suitable for input signals which can be well approximated by Laguerre functions, e.g. a few steps (Figures 4.12, 8.6 and 9.2 and report [Bjö03e]). Other types of input signals give very poor estimates.
- These methods give better time-delay estimates of systems with not too fast dynamics despite the fact these methods were derived for pure time-delay systems (Figure 8.8 and report [Bjö03e]).
- The execution time is high but can be reduced if the same input signal is used in subsequent estimations (Section 8.3.4).

Conclusions about continuous-time one-step explicit methods (Section 5.1.1):

- For a first oder model with time-delay there is at most one minimum within each sampling interval (Section 5.1.2). (There can be a single minimum in region extending over several sampling intervals if LP-filtering is performed. See [FMS96] and Section 5.1.3)
- No matter how close to the true time-delay we start the optimization, there are still cases which lead to an incorrect local minimum. Not even close to the true time-delay, the Cramer Rao lower bound is always useful. See Section 5.1.2.
- Using no prewhitening of the input signal is best (Figure 8.9). On average these methods work better for random binary input signals than for step input signals (Figure 8.10) .

Conclusions about arxstruc type methods (Section 5.2) in open loop:

- High model orders are the best for arxstruc to be able to accurately describe the system and the noise (Section 8.5.1).
- The model orders for oestruc should be 1 for the numerator to avoid ambiguity in the time-delay and be close to the true order for the denominator (Section 8.5.1).
- The prefiltered arxstruc, here called met1struc, has nearly as good estimation quality as oestruc but is much faster. If the true noise character is unchanged for repeated time-delay estimations with met1struc, the execution speed is nearly as high as for arxstruc (Sections 8.5.1-8.5.2).
- On average these methods are better for fast than for slow systems (Fig. 9.3, 8.14) and better for random binary input signals than for step input signals (Figures 9.2 and 8.14).

Conclusions about sampling methods (Section 5.3):

- It is possible to derive an explicit expression for the time-delay of a first order system with time-delay after zero-order-hold sampling (Section 5.3.3).

Conclusions about area and moment methods utilizing estimated step and impulse responses (Section 6.1) in open loop:

- Area and moment methods using estimated impulse and step responses often fail due to inaccurate estimates of the impulse and step responses (Section 8.6 and report [Bjö03b]). (An improvement is described in [Ing03] but not tested here. In [ÅH95] measured impulse and step responses are used but not here.
- Moment methods are less reliable than area methods and are not useful.
- On average these methods work better for slow systems than for fast systems (Section 8.6 and report [Bjö03b]). This is the opposite as for thresholding methods, which also use estimated impulse and step responses.

10.4 Future Work

Some interesting future work is:

- Make a better estimate of the change time in thresholding methods (Sections 4.1.1 and 4.1.3) and test maximum likelihood based detection methods.
- Continue study of methods for separating the time-delay from the dynamics. See Section 4.1.1 and 4.1.4.
- Continue study of local minima in continuous and discrete-time one-step explicit methods.
- Future work on sampling methods could be the following: Implement and test TIDEA (Section 5.3.2). Implement and test the method with exact time-delay from the sampling process (Section 5.3.3). This method uses a first order plus time-delay model structures. Derive the exact time-delay from the sampling process also for more complex model structures than first order plus time-delay, e.g. second order model with time-delay, and for first-order-hold sampling. A second order model structure as in idproc7 (Section 7.2) would probably be better for the standard benchmark (Section 7.3.1).
- A comprehensive comparison of TDE methods on real data, for example: hair dryer [Lju99, p. 525], speech communication between driver and back seat passenger in a car [Nyg03] or radar range estimation.
- Future work on applications, e.g. system identification for control; diagnosis and performance monitoring; direction of arrival estimation in signal intelligence, radar and mobile communications; and range estimation in radar.

Bibliography

[ÅH95] K. Åström and T. Hägglund. *PID Controllers: Theory, Design and Tuning.* Instrument Society of America, 2nd edition, 1995. ISBN 1-55617-516-7.

[ÅSH84] K. J. Åström, J. Sternby, and P. Hagander. Zeros of sampled systems. *Automatica*, 20(1):31–38, 1984.

[ÅW84] K. Åström and B. Wittenmark. *Computer Controlled Systems. Theory and Design.* Prentice-Hall, 1984.

[Bjö] S. Björklund. ESDATA. A Matlab toolbox for managing and analyzing data from factorial experiments. User's guide. Technical Report LiTH-ISY-R-xxxx, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden. To be published.

[Bjö02] S. Björklund. Analysis of a phase method for time-delay estimation. Technical Report LiTH-ISY-R-2467, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, October 2002.

[Bjö03a] S. Björklund. Analysis of a phase method for time-delay estimation. Short version. Technical Report LiTH-ISY-R-2539, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, September 2003.

[Bjö03b] S. Björklund. Experimental evaluation of some cross correlation methods for time-delay estimation in linear systems. Technical Report

LiTH-ISY-R-2513, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, April 2003.

[Bjö03c] S. Björklund. Experimental evaluation of some methods using simple process models for estimating continuous time-delays in open-loop. Technical Report LiTH-ISY-R-2526, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, July 2003.

[Bjö03d] S. Björklund. Experimental evaluation of some thresholding methods for estimating time-delays in open-loop. Technical Report LiTH-ISY-R-2525, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, July 2003.

[Bjö03e] S. Björklund. Experimental open-loop evaluation of some time-delay estimation methods in the Laguerre domain. Technical Report LiTH-ISY-R-2537, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, September 2003.

[Bjö03f] S. Björklund. A study of the choice of model orders in arxstruc-type methods for open-loop time-delay estimation in linear systems. Technical Report LiTH-ISY-R-2536, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, August 2003.

[BL03] S. Björklund and L. Ljung. A review of time-delay estimation techniques. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, USA, December 2003. To be published.

[BV03] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, November 2003. Will be published in November 2003, <http://www.stanford.edu/ boyd/cvxbook.html>.

[C$^+$81] G. C. Carter et al. Special issue on time delay estimation. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 29(3), June 1981. pt 2.

[Car87] G. C. Carter. Coherence and time delay estimation. *Proceedings of the IEEE*, 75(2):236–255, February 1987.

[Car93] G. Carter. *Coherence and Time Delay Estimation - An Applied Tutorial for Research, Development, Test, and Evaluation Engineers*. IEEE Press, 1993.

[CCS94] B.-S. Chen, J.-M. Chen, and S.-C. Shern. An ARMA robust system identification using a generalized lp norm estimation algorithm. *IEEE Transactions on Signal Processing*, 42(5):1063–1073, May 1994.

[CHWW99] C. Carlemalm, S. Halvarsson, T. Wigren, and B. Wahlberg. Algorithms for time delay estimation using a low complexity exhaustive search. *IEEE Transactions on Automatic Control*, 44(5):1031–1037, May 1999.

[CST] CSTB. Matlab control system toolbox, v. 4.2.1 (R11.1). The Mathworks Inc.

[DC02] K. De Cock. *Principal Angles in System Theory, Information Theory and Signal Processing.* Phd thesis , Departement Elektrotechniek, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium, 2002. ISBN 90-5682-351-5.

[EDE89] A. Elnaggar, G. A. Dumont, and A.-L. Elshafei. Recursive estimation for system of unknown delay. In *Proceedings of the 28th Conference on Decision and Control*, Tampa, Florida, USA, December 1989.

[FHJ02] J. Falk, P. Händel, and M. Jansson. Direction finding for electronic warfare systems using the phase of the cross spectral density. In *RadioVetenskap och Kommunikation (RVK)*, pages 264–268, June 2002.

[Fis99] B. R. Fischer. *System Identification in Alternative Shift Operators with Applications and Some Other Topics.* Phd thesis LTU-DT-99/18–SE, Department of Computer Science and Electrical Engineering, Luleå University of Technology, Luleå, Sweden, August 1999.

[FM99a] B. R. Fischer and A. Medvedev. L^2 time delay estimation by means of Laguerre functions. In *Proceedings of American Control Conference*, San Diego, CA, USA, June 1999.

[FM99b] B. R. Fischer and A. Medvedev. Laguerre domain estimation of time delays in narrowband ultrasonic echoes. In *14th Triennial IFAC World Congress*, pages 361–366, Beijing, China, July 1999.

[FM99c] B. R. Fischer and A. Medvedev. Time delay estimation by means of Laguerre functions. In *6th St. Petersburg Symposium on Adaptive Systems Theory*, volume 1, pages 65–68, St. Petersburg, Russia, September 1999.

[FMS91] G. Ferretti, C. Maffezzoni, and R. Scattolini. Recursive estimation of time delay in sampled systems. *Automatica*, 27(4):653–661, 1991.

[FMS96] G. Ferretti, C. Maffezzoni, and R. Scattolini. On the identifiability of the time delay with least-squares methods. *Automatica*, 32(3):449–453, 1996.

[GL97] T. Glad and L. Ljung. *Reglerteori. Flervariabla och olinjära metoder.* Studentlitteratur, Lund, Sweden, 1997. In Swedish.

[GLM01] F. Gustafsson, L. Ljung, and M. Millnert. *Signalbehandling.* Studentlitteratur, Lund, Sweden, 2001. In Swedish.

[GS94] A. Grennberg and M. Sandell. Estimation of subsample time delay differences in narrowband ultrasonic echoes using the hilbert transform correlation. *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, 41(5):588–595, September 1994.

[Gus00] F. Gustafsson. *Adaptive Filtering and Change Detection.* Wiley, 2000. ISBN 0-471-49287-6.

[GVL96] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The John Hopkins University Press, London, United Kingdom, 3rd edition, 1996.

[Hän91] E. Hänsler. *Statistische Signale. Grundlagen und Anwendungen.* Springer Verlag, 1991.

[HJ91] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis.* Cambridge University Press, 1991. ISBN 0-521-30587-X.

[Hor00] A. Horch. *Condition Monitoring of Control Loops.* Phd thesis TRITA-S3-REG-0002, Department of Signals, Sensors and Systems, Royal Institute of Technology, Stockholm, Sweden, 2000.

[HR95] A. W. Houghton and C. D. Reeve. Detection of spread-spectrum signals using the time-domain filtered cross spectral density. *IEE Proceedings - Radar, Sonar and Navigation*, 142(6):286–292, December 1995.

[HR97] A. W. Houghton and C. D. Reeve. Direction finding on spread spectrum signals using the time-domain filtered cross spectral density. *IEE Proceedings - Radar, Sonar and Navigation*, 144(6):315–320, December 1997.

[IHD00] A. J. Isaksson, A. Horch, and G. A. Dumont. Event-triggered deadtime estimation – comparison of methods. In *Preprints Control Systems 2000*, pages 209–215, Victoria, Canada, 2000.

[IHD01a] A. J. Isaksson, A. Horch, and G. A. Dumont. Event-triggered deadtime estimation - comparison of methods. Technical report, Department of Signals, Sensors & Systems, Royal Institute of Technology, Stockholm, Sweden, May 2001.

[IHD01b] A. J. Isaksson, A. Horch, and G. A. Dumont. Event-triggered deadtime estimation from closed-loop data. In *Proceedings of American Control Conference*, Arlington, VA, USA, June 2001.

[Ing03] A. Ingimundarson. *Dead-Time Compensation and Performance Monitoring in Process Control*. Phd thesis, Dep. of Automatic Control, Lund Institute of Technology, Lund, Sweden, 2003.

[Isa97] M. Isaksson. A comparison of some approaches to time-delay estimation. Master's thesis ISRN LUTFD2/TFRT–5580–SE, Dep. Automatic Control, Lund Institute of Technology, Sweden, 1997.

[JD93] D. H. Johnson and D. E. Dudgeon. *Array Signal Processing. Concepts and Techniques*. Prentice Hall, 1993. ISBN 0-13-048513-6.

[Kau54] W. H. Kautz. Transient syntesis in the time domain. *IRE Transactions on Circuit Theory*, 1(3):29–39, September 1954.

[Kay93] S. M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice-Hall, 1993. ISBN 0-13-042268-1.

[KG81] H. Kurz and W. Goedecke. Digital parameter-adaptive control of processes with unknown dead time. *Automatica*, 17:245–252, January 1981.

[KQ92] S. Kingsley and S. Quegan. *Understanding Radar Systems*. McGraw-Hill, 1992. ISBN 0-07-707426-2.

[Kur79] H. Kurz. Digital parameter-adaptive control of processes with unknown constant or timevarying dead time. In *Proceedings of the 5th IFAC Symposium on Identification and System Parameter Estimation*, pages 1187–1194, Darmstadt, Germany, 1979.

[Lam93] J. Lam. Model reduction of delay systems using Padé approximants. *International Journal of Control*, 57(2):377–391, 1993.

[LG91] L. Ljung and T. Glad. *Modellbygge och simulering*. Studentlitteratur, Lund, Sweden, 1991. In Swedish.

[Lju99] L. Ljung. *System Identification: Theory for the User*. Prentice-Hall, Upper Saddle River, N.J. USA, 2nd edition, 1999.

[Lju02] L. Ljung. Identification for control: Simple process models. In *Proceedings of the 41st IEEE Conference on Decision and Control*, pages 4652–4657, Las Vegas, Nevada, USA, December 2002.

[Lju03] L. Ljung. *System Identification Toolbox for use with* MATLAB. *Version 6*. The MathWorks, Inc, Natick, MA, 6th edition, 2003.

[LT99] X. Lai and H. Torp. Interpolation methods for time-delay estimation using cross-correlation method for blood velocity measurent. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 46(2):277–290, March 1999.

[Mat96] Mathworks, Inc., 24 Prime Park Way, Natick, MA 01760-1500, USA. *MATLAB Control System Toolbox. User's Guide*, December 1996.

[Mat98] Mathworks, Inc., 24 Prime Park Way, Natick, MA 01760-1500, USA. *MATLAB Higher-Order Spectral Analysis Toolbox. User's Guide*, January 1998.

[Mat01] The MathWorks, Inc. *MATLAB Statistics Toolbox. User's Guide. Version 3*, 2001.

[MG$^+$98] H. Messer, J. Goldberg, et al. Highlights of statistical signal and array processing. *IEEE Signal Processing Magazine*, pages 21–64, September 1998.

[Mod91] Moddemeijer. On the determination of the position of extrema of sampled correlators. *IEEE Transactions on Signal Processing*, 39(1):216–219, January 1991.

[Mon97] D. C. Montgomery. *Design and Analysis of Experiments.* Wiley, 1997. ISBN 0-471-15746-5.

[MZJ87] M. Malek-Zavarei and M. Jamshidi. *Time-Delay Systems. Analysis, Optimization and Applications.* Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1987. ISBN 0-444-70204-0.

[NL91] P. Nagy and L. Ljung. Estimating time-delays via state-space identification methods. In *Preprints 9th IFAC Symposium on System Identification and System Parameter Estimation*, pages 1141–1144, Budapest, Hungary, Jul 1991.

[NM93] C. L. Nikias and J. M. Mendel. Signal processing with higher-order spectra. *IEEE Signal Processing Magazine*, pages 10–37, July 1993.

[NP88] C. L. Nikias and R. Pan. Time delay estimation in unknown gaussian spatially correlated noise. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(11):1706–1714, November 1988.

[NS95] D. T. Nagle and J. Saniie. Performance analysis of linearly combined order statistic cfar detectors. *IEEE Transactions on Aerospace and Electronic Systems*, 31(2):522–533, April 1995.

[Nyg03] M. Nygren. Improved speech communication in a car. Master's thesis LiTH-ISY-EX-3397-2003, Department of Electrical Engineering, Linköping University, Linköping, Sweden, June 2003.

[Pag54] E. S. Page. Continuous inspection schemes. *Biometrika*, 41:100–115, 1954.

[Pup85] R. Pupeikis. Recursive estimation of the parameters of linear systems with time delay. In *Proc. 7th IFAC/IFORS Symp. Identification and System Parameter Estimation*, pages 787–792, York, UK, July 1985.

[Qua81] A. H. Quazi. An overview on the time delay estimate in active and passive systems for target localization. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 29(3):527–533, June 1981. pt. 2.

[SBS97] A. Soumelidis, J. Bokor, and F. Schipp. Representation and approximation of signals and systems using Generalized Kautz Functions. In *Proceedings of the 36th Conference on Decision and Control*, pages 3793–, San Diego, California, USA, December 1997.

[SIT] SITB. Matlab system identification toolbox v. 5.0.1 (R12.1). The Mathworks Inc.

[Sum95] A. Sume. Kursmaterial i radarteori. Technical Report FOA-D–95-00108-3.3-SE, Department of Sensor Technology, Swedish Defence Research Establishment, Box 1165, 581 11 Linköping, Sweden, April 1995. In Swedish.

[Swa99] A. P. Swanda. *PID Controller Performance Assessment Based on Closed-Loop Response Data*. Phd thesis, University of California, Santa Barbara, California, USA, June 1999.

[Wah91] B. Wahlberg. System identification using Laguerre models. *IEEE Transactions on Automatic Control*, AC-36(5):551–562, May 1991.

[Wah94] B. Wahlberg. System identification using Kautz models. *IEEE Transactions on Automatic Control*, 39(6):1276–1282, June 1994.

[Wik02] M. Wikström. Utveckling och implementering av ett audiopejlsystem baserat på tidsdifferensmätning. Master's thesis LiTH-ISY-EX-3277-2002, Linköpings universitet, Linköping, Sweden, October 2002. In swedish.

[WZ01] Q.-G. Wang and Y. Zhang. Robust identification of continuous systems with dead-time from step responses. *Automatica*, pages 377–390, 2001.

A

ANOVA and Confidence Interval Validation

For the ANOVA and confidence intervals used in this thesis to be applicable some prerequisites must be fulfilled (Section 7.5.3). This appendix contains validation graphs to determine if the prerequisites are fulfilled. It also contains ANOVA tables (Section 7.5.3). The sub-appendices appear in the order of the corresponding analyses in the main part of this thesis.

A.1 Validation of ANOVA and Confidence Intervals for Time Domain Methods

A.1.1 Choosing parameters of direct thresholding of impulse response

This appendix contains the ANOVA table (Table A.1), transformation graph (Figure A.1) and validation graphs (Figures A.2-A.3) for the analysis in Section 8.1.1.

A.1.2 Properties of direct thresholding of impulse response

This appendix contains the ANOVA table (Table A.2), transformation graph (Figure A.4) and validation graphs (Figures A.5-A.6) for the analysis in Section 8.1.2.

Figure A.1 Plot for choosing a variance-stabilizing transform [Mon97] for ANOVA for choosing parameters in the method idimp4 (Section 8.1.1). (t228b4.m)

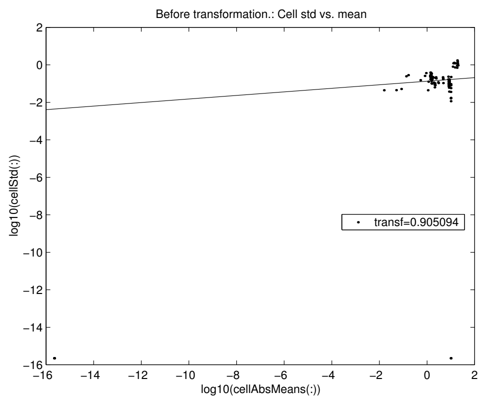


Figure A.2 Residual analysis for ANOVA and confidence intervals for choosing parameters in the method idimp4 (Section 8.1.1). (t228b4.m)

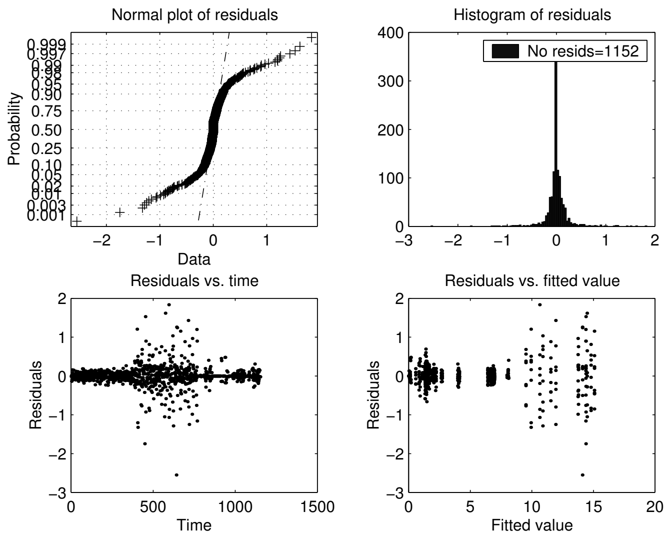


Table A.1 Parts of the ANOVA table for choosing parameters in the method idimp4 (Section 8.1.1). Constrained (Type III) sums of squares [Mat01]. (t228b4.m)

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
InType	7132.2532	2	3566.1266	46034.6821	0
Prewhite	0.29015	1	0.29015	3.7455	0.053229
SNR	590.9226	1	590.9226	7628.1459	0
Sys	379.4393	3	126.4798	1632.7117	0
threshold	2756.3471	2	1378.1736	17790.6701	0
...					
Prewhite*threshold	0.13934	2	0.069672	0.89938	0.40715
...					
Error	78.0858	1008	0.077466		
Total	17994.5698	1151			

Table A.2 Parts of the ANOVA table for studying properties of the method idimp4 (Section 8.2.1). All values in the column "Prob>F" are zero. Constrained (Type III) sums of squares [Mat01]. (t229b5)

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
Method	55.7444	3	18.5815	2080.7382	0
InType	4244.999	2	2122.4995	237676.0025	0
SNR	850.0346	1	850.0346	95186.2773	0
Sys	186.3467	3	62.1156	6955.6581	0
Method*InType	124.0827	6	20.6804	2315.7814	0
Method*SNR	26.6615	3	8.8872	995.1773	0
Method*Sys	44.479	9	4.9421	553.4135	0
InType*SNR	719.7577	2	359.8789	40298.9837	0
...					
Error	2.5719	288	0.0089302		
Total	6998.4628	383			

A.2 Properties of DAP Methods

This appendix contains the ANOVA table (Table A.3), transformation graph (Figure A.4) and validation graphs (Figures A.5-A.6) for the analysis in Section 8.1.2.

Figure A.3 Residual standard deviation versus factor levels for ANOVA and confidence intervals for choosing parameters in the method idimp4 (Section 8.1.1). (t228b4.m)

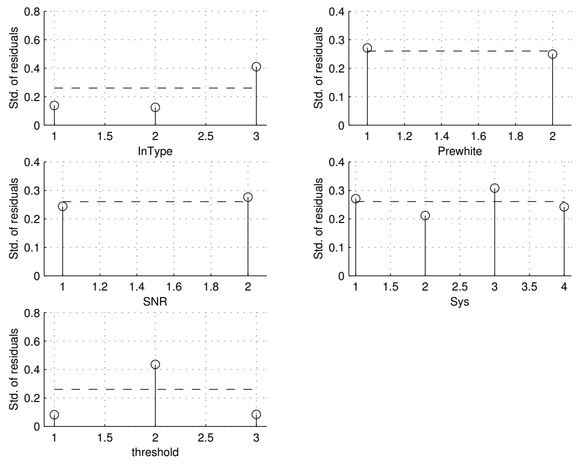


Table A.3 Parts of the ANOVA table for studying properties of DAP methods (Section 8.2.2). Constrained (Type III) sums of squares [Mat01]. (t208b16)

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
Method	11.1833	3	3.7278	1830.486	0
InType	21.562	2	10.781	5293.8965	0
Sys	4.6106	3	1.5369	754.6695	0
Method*InType	2.6205	6	0.43676	214.4654	0
Method*Sys	3.6425	9	0.40473	198.7364	0
InType*Sys	4.6853	6	0.78089	383.4469	0
Method*InType*Sys	2.1486	18	0.11936	58.6128	0
Error	0.29326	144	0.0020365		
Total	50.7462	191			

A.3 Validation of ANOVA and Confidence Intervals for Laguerre Domain Methods

A.3.1 Several Laguerre domain approximation methods with only steps

In this section we show validation graphs for the ANOVA and confidence intervals in Section 8.3.2. The used transformation [Mon97] is depicted in Figure A.10. We see in Figures A.11-A.12 that the prerequisites are approximately fulfilled [Mon97].

Figure A.4 Plot for choosing a variance-stabilizing transform [Mon97] for ANOVA for studying properties of the method idimp4 (Section 8.1.2).

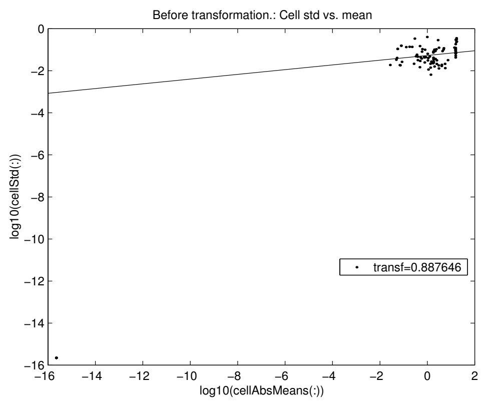


Figure A.5 Residual analysis for ANOVA and confidence intervals for studying properties of the method idimp4 (Section 8.1.2).

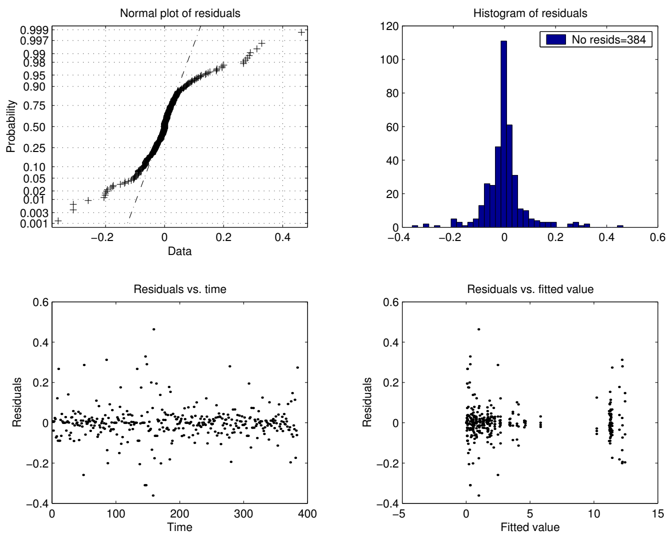


Figure A.6 Residual standard deviation versus factor levels for ANOVA and confidence intervals for studying properties of the method idimp4 (Section 8.1.2).

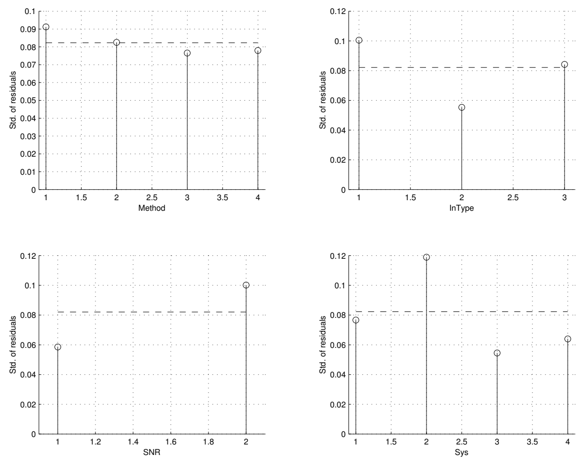


Table A.4 Parts of the ANOVA table for several Laguerre domain approximation methods (Section 8.3.2) with input signal Steps. SNR = 1. Constrained (Type III) sums of squares [Mat01]. (t224b1)

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
Method	2769.8838	7	395.6977	9274.3874	0
Sys	84.5622	3	28.1874	660.6579	0
Method*Sys	115.7668	21	5.5127	129.2071	0
Error	4.0959	96	0.042666		
Total	2974.3087	127			

A.3.2 The method fischer8 with step input signals at low SNR

In this section we show validation graphs for the ANOVA and confidence intervals in Section 8.3.3. The used transformation [Mon97] is depicted in Figure A.13. We see in Figures A.14-A.13 that the prerequisites are approximately fulfilled [Mon97].

A.4 Validation of ANOVA and Confidence Intervals for Idproc Methods

This appendix comments on the use and applicability of the ANOVA and confidence intervals in Section 8.4. The ANOVA table is given in Figure A.6 and the transformation graph in Figure A.15. We see in validation graphs (Figures A.16-A.17)

Figure A.7 Plot for choosing a variance-stabilizing transform [Mon97] for ANOVA for studying properties of DAP methods (Section 8.2.2). (t208b16)

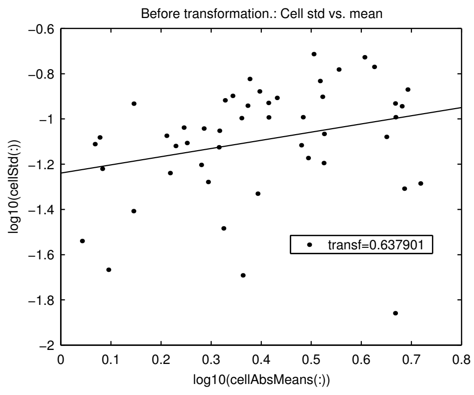


Figure A.8 Residual analysis for ANOVA and confidence intervals for studying properties of DAP methods (Section 8.2.2). (t208b16)

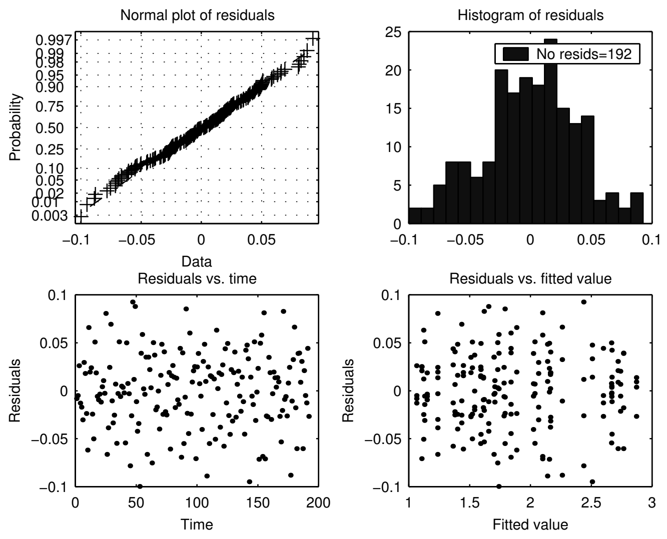


Figure A.9 Residual standard deviation versus factor levels for ANOVA and confidence intervals for studying properties of DAP methods (Section 8.2.2). (t208b16)

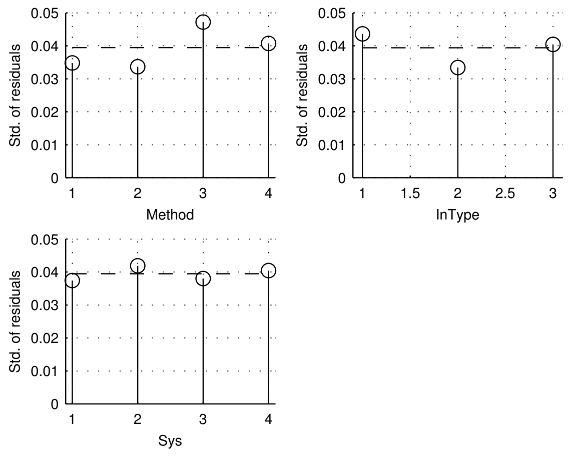


Figure A.10 Plot for choosing a variance-stabilizing transform [Mon97] for ANOVA for several Laguerre domain approximation methods (Section 8.3.2) with input signal Steps. SNR = 1. Transformation: (RMS error)$^{0.96}$. (t224b1)

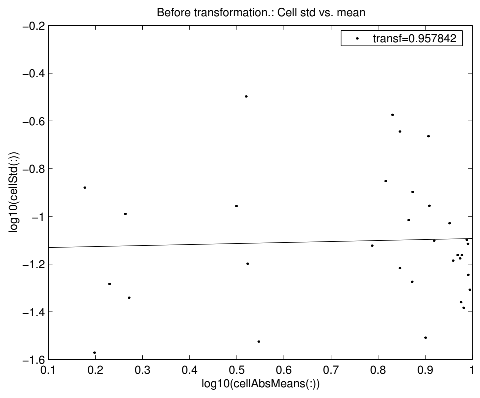


Table A.5 The ANOVA table for fischer8 with input Stepsand SNR=1 (Section 8.3.3). Constrained (Type III) sums of squares [Mat01]. (t225b1)

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
Sys	380.3476	4	95.0869	2771.8578	0
Error	0.51457	15	0.034304		
Total	380.8622	19			

Figure A.11 Residual analysis for ANOVA and confidence intervals for several Laguerre domain approx. methods (Section 8.3.2) with input signal Steps. SNR = 1. (t224b1)

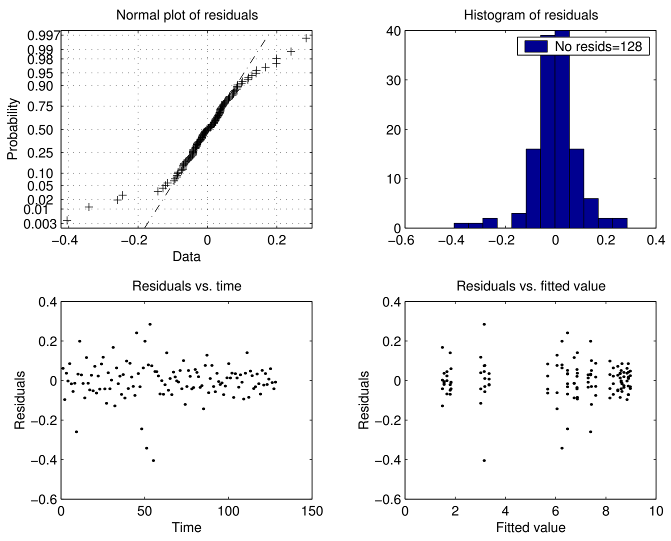


Figure A.12 Residual standard deviation versus factor levels for ANOVA and confidence intervals for several Laguerre domain approximation methods (Section 8.3.2) with input signal Steps. SNR = 1. (t224b1)

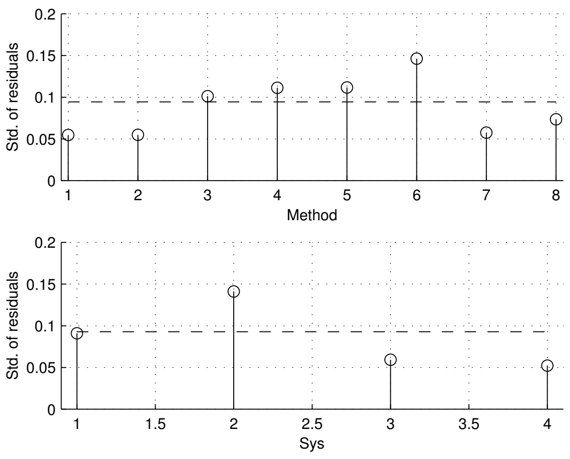
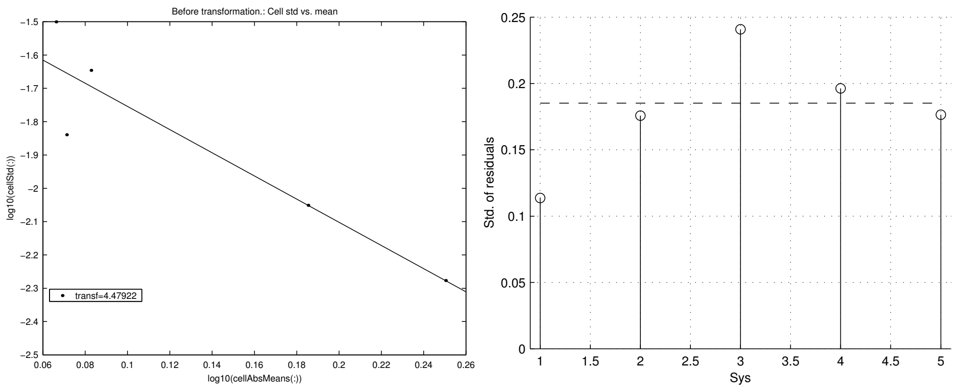


Figure A.13 Transform and residual standard deviation for fischer8 with input Stepsand SNR=1 (Section 8.3.3). (t225b1)



a. Plot for choosing a variance-stabilizing transform.

b. Residual standard deviation vs. factor levels.

Figure A.14 Residual analysis for ANOVA and confidence intervals for fischer8 with input signal Stepsand SNR = 1 (Section 8.3.3). Transformation: $(\text{RMS error})^{4.48}$. (t225b1)

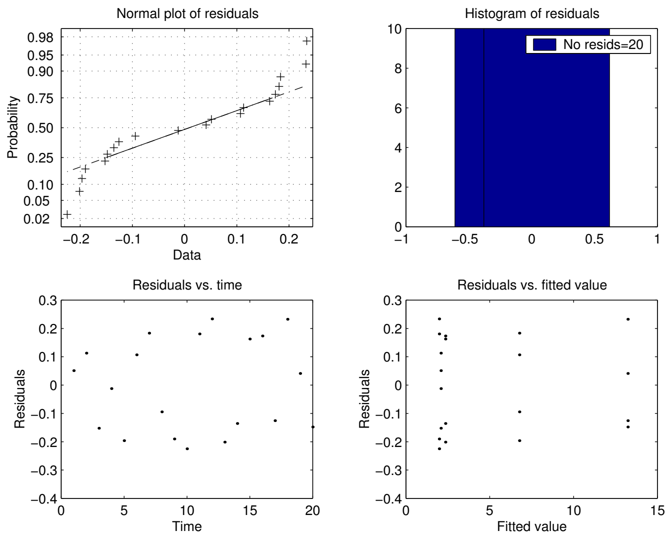


Table A.6 Parts of the ANOVA table for all idproc methods (Section 8.4). Constrained (Type III) sums of squares [Mat01]. (t233b2)

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
Method	0.046813	4	0.011703	377.6928	0
InType	0.076806	2	0.038403	1239.3547	0
Prewhite	0.0075127	1	0.0075127	242.4505	0
SNR	0.038715	1	0.038715	1249.4163	0
Sys	0.01284	3	0.0042801	138.1277	0
Method*InType	0.013451	8	0.0016813	54.2596	0
Method*Prewhite	0.00054883	4	0.00013721	4.428	0.00153
Method*SNR	0.0032563	4	0.00081407	26.2717	0
Method*Sys	0.027518	12	0.0022932	74.0064	0
...					
Method*InType*Prewhite	0.0014093	8	0.00017617	5.6853	5.0237e-07
...					
Method*Prewhite*Sys	0.00069846	12	5.8205e-05	1.8784	0.033792
...					
Error	0.02231	720	3.0986e-05		
Total	0.29706	959			

that the prerequisites are not completely fulfilled so we must be somewhat careful in the interpretation of the ANOVA and confidence interval. See [Mon97, Bjö03b] for more information on this.

A.5 Validation of ANOVA and Confidence Intervals for Arxstruc Type Methods

This appendix contains the ANOVA table (Table A.7), transformation graph (Figure A.18) and validation graphs (Figures A.19-A.20) for the analysis in Section 8.5.2.

A.6 Validation for Promising Methods with Only Steps and SNR =1

In this section we show validation graphs for the ANOVA and confidence intervals in Section 9.1.2 (SNR=1). The ANOVA table is given in Figure A.8 and the transformation graph in Figure A.21. We see in Figures A.22-A.23 that the prerequisites are approximately fulfilled.

Figure A.15 Plot for choosing a variance-stabilizing transform [Mon97] for ANOVA of all idproc methods (Section 8.4).

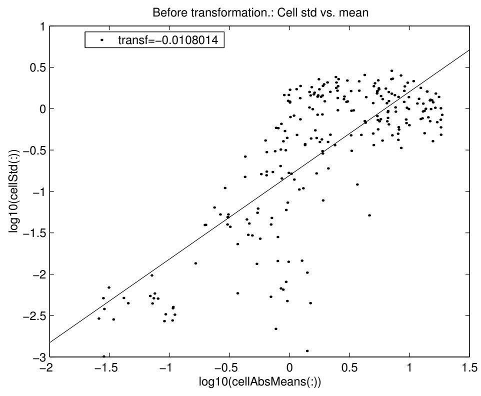


Figure A.16 Residual analysis for ANOVA and confidence intervals for all idproc methods (Section 8.4).

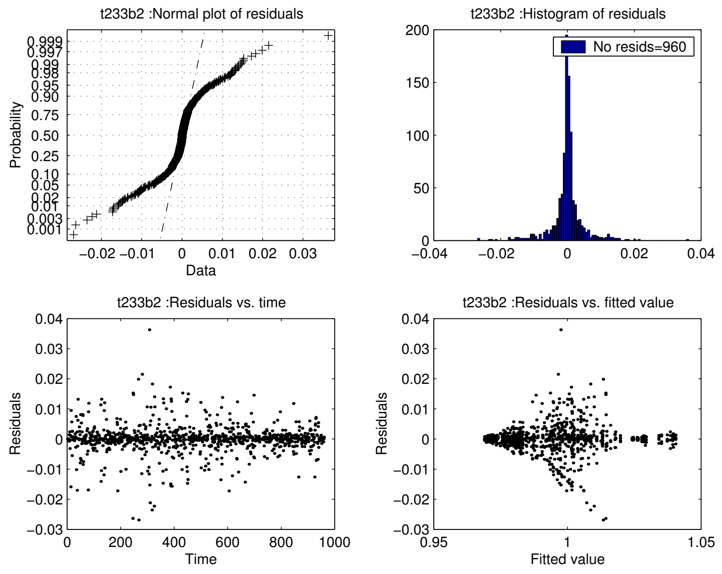


Figure A.17 Residual standard deviation versus factor levels for ANOVA and confidence intervals for all idproc methods (Section 8.4).

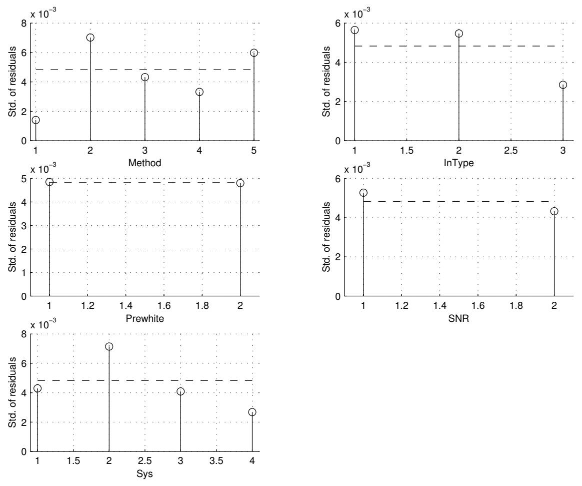


Table A.7 ANOVA table for properties of Arxstruc type methods (Section 8.5.2). Constrained (Type III) sums of squares [Mat01]. (t208b15)

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
Method	18.2507	2	9.1254	63394.725	0
InType	38.8245	2	19.4122	134858.6174	0
Sys	2.8725	3	0.95752	6651.954	0
Method*InType	5.8451	4	1.4613	10151.5517	0
Method*Sys	34.2838	6	5.714	39695.4566	0
InType*Sys	4.6773	6	0.77954	5415.5501	0
Method*InType*Sys	18.9597	12	1.58	10976.2392	0
Error	0.015546	108	0.00014395		
Total	123.7291	143			

Table A.8 The ANOVA table for some promising methods with input signal Steps in open loop with SNR = 1 (Section 9.1.2). Constrained (Type III) sums of squares [Mat01]. (t208b10)

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
Method	5.541	5	1.1082	815.1808	0
Sys	4.0479	3	1.3493	992.5378	0
Method*Sys	2.6613	15	0.17742	130.5073	0
Error	0.097881	72	0.0013595		
Total	12.3481	95			

Figure A.18 Plot for choosing a variance-stabilizing transform [Mon97] for ANOVA of properties of arxstruc type methods (Section 8.5.2). (t208b15)

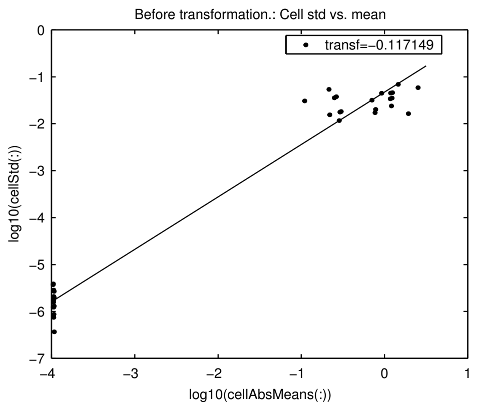


Figure A.19 Residual analysis for ANOVA and confidence intervals for properties of arxstruc type methods (Section 8.5.2). (t208b15)

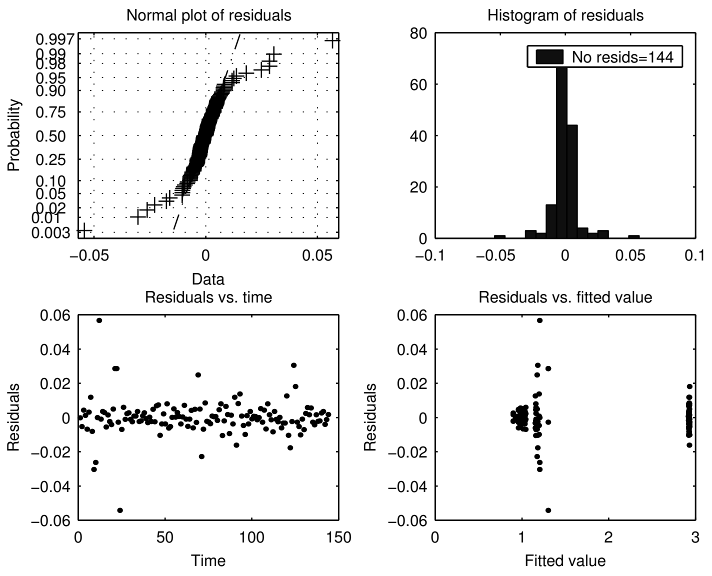


Figure A.20 Residual standard deviation versus factor levels for ANOVA and confidence intervals for properties of arxstruc type methods (Section 8.5.2). (t208b15)

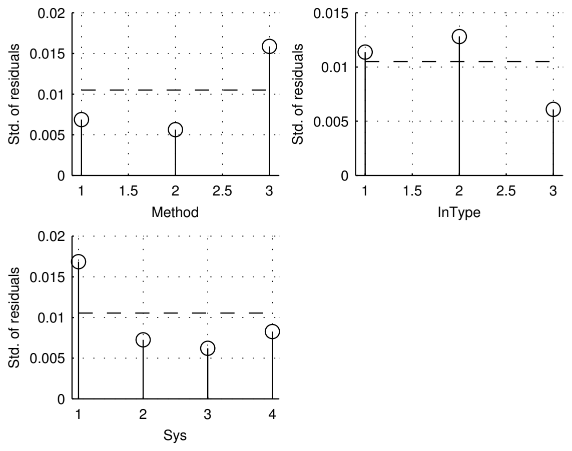


Figure A.21 Plot for choosing a variance-stabilizing transform for ANOVA for some promising methods with input signal Steps in open loop with SNR = 1 (Section 9.1.2). Transformation: (RMS error)$^{0.48}$. (t208b10)

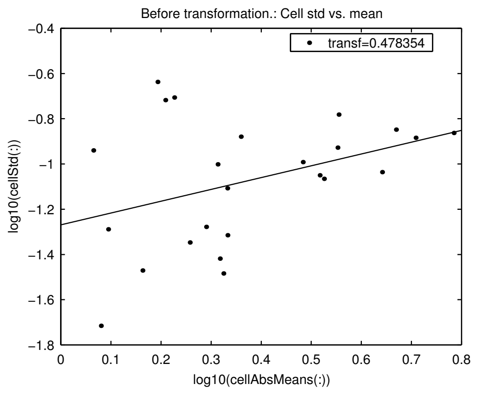


Figure A.22 Residual analysis for ANOVA and confidence intervals for some promising methods with input signal Steps in open loop with SNR = 1 (Section 9.1.2). (t208b10)

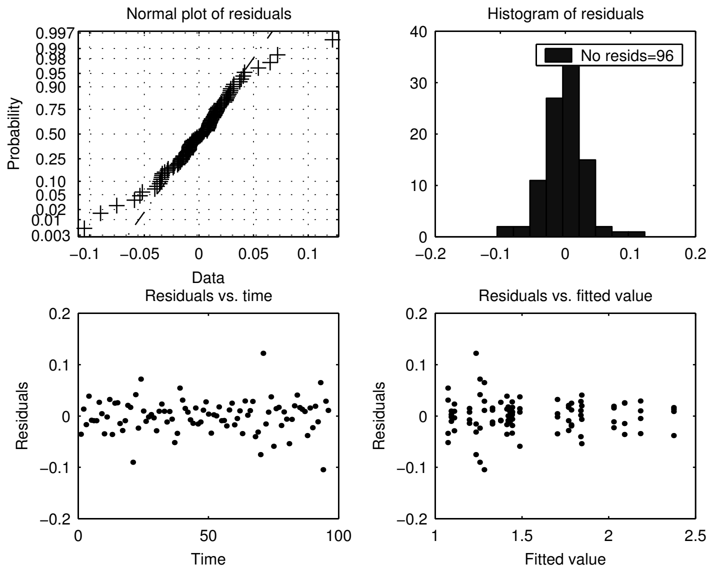
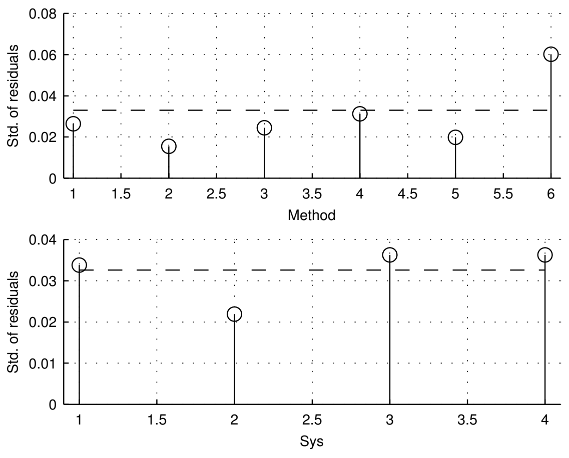


Figure A.23 Residual standard deviation versus factor levels for ANOVA and confidence intervals for some promising methods with input signal Steps in open loop with SNR = 1 (Section 9.1.2). (t208b10)



Tekn. lic. Dissertations
Division of Automatic Control and Communication Systems
Linköping University