Algorithms for estimation of model errors and uncertainties arising in black box system identification

van Riel, N.A.W.

Award date:
1995

Technische Universiteit Eindhoven

Faculty of Electrical Engineering
Measurement & Control group

Master's Thesis:

Algorithms for estimation of model errors and uncertainties arising in black box system identification

by N.A.W. van Riel

Coach : ir. L.J.J.M. Ariaans
Supervisor : Prof. dr. ir. P.P.J. van den Bosch
Period : January 1995 - October 1995

Summary

van Riel, N.A.W.; 'Algorithms for estimation of model errors and uncertainties arising in black box system identification'

Master's Thesis, Measurement and Control group, department of Electrical Engineering, Eindhoven University of Technology, October 1995.

To make a system behave like one would like to, a controller is needed. To design a suitable controller, the relevant dynamics of the system have to be known, i.e. a model has to be identified. For more complex systems, it is often easier and faster to use so called 'Black Box' modelling (a computer algorithm calculates a model using input-output data of the system), than to combine all kinds of physical equations. To develop good controllers it is useful to have a certain notion of the quality of the models, i.e. the model error should be known. The Measurement & Control group of the department of Electrical Engineering of Eindhoven University of Technology was interested in an evaluation of the algorithms which have appeared in literature about this subject. Also attempts have been made to implement certain useful algorithms in Matlab.
In 'Black Box' identification two areas can be distinguished. The approach which is often referred to as 'traditional', takes a *stochastic* problem formulation. An algorithm is presented which specifies the model error in frequency domain for this stochastic approach. However, this algorithm only holds for FIR models and is approximative for ARX models. There are two versions, of which one is a continuous algorithm which is difficult to implement and the other algorithm uses optimization, for which generally convergence to the true optimal result is not guaranteed. The algorithm has been implemented in Matlab in the way it is presented in literature. However, in this form the algorithm is not suitable for general applications.
More recent Modern (Robust) control techniques are capable of constructing 'optimal' controllers, if an upperbound of the model error is known. A *deterministic* approach of identification can be made more in agreement with this. A problem formulation of identification is presented which yields models together with the corresponding upperbound for the model error. For this problem several algorithms are presented, both linear and nonlinear, which all have a two-stage structure. However these algorithms yield models of very high order. The theoretical background has been examined and this approach offers good prospects. A start has been made with the implementation of these algorithms in Matlab.

Samenvatting

Om een systeem zich te laten gedragen zoals gewenst, is er een regelaar nodig. Om een geschikte regelaar te kunnen ontwerpen, moet de relevante dynamica van het systeem bekend zijn, m.a.w. er moet een model geïdentificeerd worden. Voor complexere systemen is het vaak eenvoudiger en sneller om gebruik te maken van zogenaamde 'Black Box' modellering (een computer algoritme berekent een model aan de hand van input-output data van het systeem), dan om allerlei fysische systeemvergelijkingen aan elkaar te koppelen. Om goede regelaars te ontwikkelen is het nuttig om enig idee over de kwaliteit van de modellen te hebben, de modelfout zou bekend moeten zijn. De vakgroep Meet & Regeltechniek van de faculteit Elektrotechniek van de Technische Universiteit Eindhoven was geïnteresseerd in een evaluatie van de algoritmen die in de literatuur verschenen zijn over dit onderwerp. Ook zijn pogingen ondernomen om bepaalde nuttige algoritmen in Matlab te implementeren.

In de 'Black Box' identificatie kunnen twee stromingen onderscheiden worden. De benadering, die vaak 'traditioneel' genoemd wordt, neemt een *stochastische* probleem formulering. Een algoritme wordt behandeld dat de modelfout in frequentie domein vastlegt voor deze stochastische benadering. Het nadeel is dat dit algoritme alleen geldt voor FIR modellen en een benadering wordt toegepast voor ARX modellen. Verder zijn er twee versies, waarvan één een continu algoritme is dat moeilijk geïmplementeerd kan worden en het andere algoritme maakt gebruik van optimalisering, hetgeen in het algemeen niet garandeert dat het resultaat ook daadwerkelijk optimaal is. Het algoritme is geïmplementeerd in Matlab op de manier zoals het in de literatuur te vinden is. In deze vorm is het algoritme echter niet geschikt voor algemene toepassing.
De recentere Moderne (Robuuste) regeltechnieken kunnen in principe 'optimale' regelaars construeren, mits de modelfout afgebakend is. Een *deterministische* benadering van identificatie kan hiervoor geschikter zijn. Er wordt een probleemformulering voor identificatie behandeld die modellen oplevert samen met de bijbehorende bovengrens voor de modelfout. Voor dit probleem worden verschillende algoritmen behandeld, zowel lineair en niet-lineair, welke allen een twee-stadia structuur hebben. Deze algoritmen leveren echter wel modellen van erg grote orde op. De theoretische achtergrond is bestudeerd en deze benadering biedt goede perspectieven. Er is een eerste aanzet gegeven aan de implementatie van deze algoritmen in Matlab.

Preface

In September 1991 I have started the study Electrical Engineering at Eindhoven University of Technology. Now I am at the point of finishing this period of my life. During the study programme I've become interested in Measurement & Control, mainly because this part of Electrical Engineering is not specialistic. A mathematical and abstract way of looking at all kinds of processes is developed and this insight can be used in many technical areas.
After a nice practical training with the Measurement & Control group of the faculty of Electrical Engineering of the TUE, I have decided to do my Master's Thesis with the same group. After a very practical subject for my training, I have chosen a more abstract and theoretic project for my Master's Thesis. This report gives an account of the work I have done in the 9 months after the start in January 1995. I hope my research and software implementation can contribute to the knowledge about theoretical system identification within 'our' group. I would like to thank everybody who had co-operated in any way and in particular I would like to thank my direct coach ir. L. Ariaans who has been very supporting and has had many clarifying ideas when I was at a dead end.

Table of contents

Appendices

1. Introduction and problem definition

1.1 A general introduction

Since man became conscious of time and noticed he had a past and there was a future, he has taken great pains over controlling his environment and his future. In the early ages these attempts to influence life contained a lot of magic and later on the church played a major role. Since the Renaissance we have put an increasing amount of faith in our ratio and in the accompanying science. After the Industrial Revolution, when man has kept on inventing all kinds of new machines to relieve his daily work, the importance of ways to control these technical processes increased. First, people were the overall controllers of the processes in for example a factory, but with the increasing complexity and increasing pays this was no longer possible. Man started a search for technical expedient resources to ease his controlling job. During the Mechanization period this mainly involved the development of ways to measure things in processes. Man could use this information for better control of the process. When the Automation started, man gave away a part of his most easy controlling tasks to technical expedient resources, which were often electrical circuits. Nowadays processes are controlled by electronics (read 'computers') at much higher levels and this development still goes on.
In order to develop controllers for a process some information is necessary about (the dynamics of) the process. In other words, the system has to be identified and has to be put in some kind of model. The kind of model depends on the kind of controller which one wants to implement. Another application of models is process analysis: try to predict the behaviour of a process in certain circumstances.
One way to construct a model is to use the physical equations which belong to the components of the process. After the equations have been determined, the system can be represented by several interconnected boxes with equations in it. These equations describe the transfer from an input of the box to the output. This is called 'White Box Modelling'. Of course this model never describes the system exactly. The physical equations describe a simplified situation and only hold under certain assumptions and conditions. For more complex systems, this White Box Modelling is very time-consuming or even impossible. During the last few decades a new way of system identification has been developed: 'Black Box Identification'. The different inputs of the process are excited with some kind of (time) signal and the outputs are measured. These data are put into a computer algorithm which calculates a certain type of model. The resulting model has no direct relation to the physical reality, therefore the name '*Black* Box Modelling'. Also these models do not completely

describe a process and have limited validity. For example the resulting models are usually linear and of a finite (small) order although almost all processes are nonlinear and of high order. Often a system is only identified for certain frequencies or range of frequencies and so the model is only valid for or 'near' these working points. During measurement and identification all kinds of noise appear which deteriorate the identification: noise in the sensors, quantization errors in the computer algorithms etc.

Reduction in uncertainty is the primary role of system identification in control system design. This can be done by using more accurate modelling techniques (such as non-linear models), or by determination of the quality of the limited model. What are the errors with a certain input, in which frequency ranges are these errors small and where large etc.? This information can be used to develop controllers, especially Modern / Robust controllers. So this idea matches the framework and assumptions underlying Modern Robust control design techniques (*Control Oriented System Identification methods*). Robust controllers can, at least in principle, achieve a better performance than more classical controllers by making explicit use of prior information about the model error. It is useful if, besides verification of the model with the true system, also mathematical methods for the determination of model errors and uncertainties are available.

Two approaches of Black Box identification can be distinguished. The 'traditional' approach is to assume that the system can be described by one single model, so the system is in the set of models which is considered. The noise is not fixed, but is described as a stochastic quantity. This is why this is called a *stochastic approach* of system identification. Ljung is the name which is most often associated with the stochastic identification. See [Ljung 87] and also [Stochastische Systeemtheorie].
All other identification methods, which do not use the stochastic setting are called *deterministic*. The resulting models are usually *non-parametric* and often of high order. Most recent developments in system identification methods concentrate on deterministic approaches which are strongly related to Robust control in the way that these algorithms yield models with an upperbound for the model error.

1.2 Problem definition of the Master's Thesis

Early attempts to attach a quality tag to the traditional, parametric models (see chapter 3) are described in for example [Stochastische Systeemtheorie]. These are all *validation* techniques which are based on the attempt to show that the model is invalid. These do not yield actual error functions or bounds and are therefore not suitable for Robust controller synthesis.
From the end of the 80's articles began to appear about methods to estimate model errors and uncertainties. This is likely to be an important development for the near future of Measurement & Control science and so in the Measurement & Control group of the department of Electrical Engineering of Eindhoven University of Technology an interest has grown in these algorithms. The people in this group who deal with theoretical system identification would like to have an overview of the literature and an evaluation of the corresponding algorithms to know if these could be useful in practice and to compare them with their own work. This became the problem definition of this Master's Thesis.

The goal of this Master's Thesis is:

> *To collect different algorithms for the estimation of model errors and uncertainties from literature, to evaluate these algorithms and to implement the suitable methods in the mathematical software package Matlab.*

This report concentrates on a stochastic approach which is based on the algorithms of Goodwin & Salgado [Goodwin 89] and Goodwin & Ninness [Goodwin 92]. Starting with the available stochastic identification techniques, which were not developed with the idea to have the model error available, an attempt is made to find expressions for the model error in frequency domain. This approach will be treated in chapter 2 and 3.
Deterministic algorithms which identify a model and simultaneously determine the worst case model error are based on the problem formulation of Helmicki, Jacobson & Nett. Algorithms for this approach can be found in [Helmicki 90a], [Helmicki 90b], [Helmicki 91], [Gu 92], [Mäkilä 92], [Akçay 93] and [Jacobson 93]. This type of algorithm will come up for discussion in chapter 6 and 7.
These two different approaches are implemented in Matlab. For the algorithms of Goodwin this appears to be quite troublesome. This implementation is discussed in chapter 4. Some notes on the implementation of the deterministic algorithms can be found in chapter 7. In chapter 5 the implemented algorithms of Goodwin will be tested in some examples, among other things with a simulation of a practical process. Finally, chapter 8 contains the conclusions and recommendations for further research.

2. Notation and definitions for a stochastic embedding

2.1 Notation

N: the number of *uniformly spaced* experimental data samples in time domain (N is finite)
$\mathbb{R}$, $\mathbb{C}$, Z: the sets of real and complex numbers and integers respectively
$Z_+ := \{k \in Z: k > 0\}$, the positive integers
$\mathbb{C}^n$: space of n dimensional complex vectors
$\mathbb{C}_\infty^n$: normed space given by $\mathbb{C}^n$ together with the norm $\|f\|_\infty := \max_k |f_k|$
$\mathbb{C}_{\sigma+}$: complex numbers of which the real part is bigger than some real number σ ($Re(s) > \sigma$)

An asterisk as superscript (*) with a vector or matrix denotes conjugate, the superscript T (^{T}) denotes transposed and ^{*T} means conjugate transposed.
$\mathcal{E}\{.\}$ denotes expectation.
The covariance function of a certain quantity or function $X(\omega)$ is $cov\{X(\omega_1,\omega_2)\} := \mathcal{E}\{X(\omega_1)X(\omega_2)^*\}$, a scalar function.

ρ_t is the derivative operator in time domain. So $\rho_t^k x(t)$ is the k-th derivative of x with respect to t. $P^k(\rho_t)$ is a vector of derivative operators ρ_t from the k-th derivative operator down to the 0-th order derivative operator. When this vector is transformed to the frequency space with the Fourier transform then $P^k(j\omega)$ is a vector of decreasing powers of $j\omega$.

2.2 The identification and control environment

2.2.1 Class of systems under consideration

The class of systems to which the unknown system will be assumed to belong is the class of *stable*, *SISO* (Single Input Single Output), *LTI* (Linear Time Invariant) and *causal* systems. Remark: in literature often Linear Shift Invariant (LSI) is used which is said to be a little more general than LTI, but often a good definition of LSI lacks.

2.2.2 Transfer function

In the time domain the system is described by the convolution y = g*u, i.e.

$$y(k) = \sum_{l=0}^{\infty} g(l)u(k-l)$$

where y(k) is the system output, u(k) is the input and g(k) is the impulse response of the true system.

Often the q operator is used instead of the Z-transform:
- the *forward shift operator* q by qu(k) = u(k+1) and
- the *backward shift operator* q^{-1} by $q^{-1}u(k) = u(k-1)$.

This is done since the class of systems which is considered in literature is often 'LSI' and the Z-transform is defined for LTI. In the embedding as presented here, there is no difference. With this, the convolution can be rewritten as

$$y(k) = \sum_{l=0}^{\infty} g(l)[q^{-l}u(k)] = \left[\sum_{l=0}^{\infty} g(l)q^{-l}\right]u(k).$$

The corresponding transfer function is

$$G(q) = \sum_{l=0}^{\infty} g(l)q^{-l} \qquad (l=0,1,...).$$

In the 'shift-operator domain' this convolution becomes a multiplication: Y(q) = G(q)U(q). The frequency-domain equivalent of the transfer function G(q) is the frequency function and it is obtained by replacing q by $e^{j\omega}$.

2.2.3 The output of the true system

It is assumed that the observed data are generated by a system with a *Finite Impulse Response* (FIR) model structure:

$\mathbf{y_k = G_T(q)u_k + H(q)e_k = G_T(q)u_k + v_k}$ (2.1).

The noise filter H(q) is a rational transfer function and is assumed to be strictly stable (no poles in $|q| \geq 1$). This also holds for the true system, but this is trivial since $G_T(q)$ can be described by a FIR.

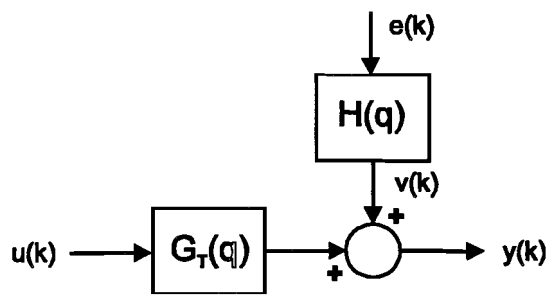

figure 2.1: The true system

This model structure is visualised in figure 2.1.

The stochastic embedding of the system identification appears in two ways:
- the noise e_k is described as a stochastic process; it is assumed that e_k is *Zero Mean White*

Noise (ZMWN), independent of input u_k and so by filtering with H the output noise v_k is coloured;

- also the true system G_T is assumed to be a specific realisation of a stochastic process, described by the input-output data; this is a renewal of the 'traditional' approach of identification.

Remark: - Zero Mean means $\mathcal{E}\{e_k\} = 0$ and White means $\mathcal{E}\{e_k e_l^T\} = 0$ if $k \neq l$ and $\mathcal{E}\{e_k e_l^T\} = \sigma^2$ if $k = l$.

3. Model errors in the stochastic approach of system identification

In this chapter algorithms will be derived for computation of model errors which arise in so called *parametric identification* in a stochastic approach of system identification. The background of this stochastic parametric identification can be found in [Ljung 87] and [Stochastische Systeemtheorie]. The identification algorithms use a *prediction model*: calculate a future output of an (initial) model and compare this with the actual output. A minimization problem has to be solved: a model structure is chosen and the parameters in the structure are obtained by minimizing a *Cost Function*, which is a function of the *prediction error* (the difference between the true and the estimated output). This error has to be small for a good model (i.e. the cost function is small). The resulting model is a *Transfer Function* or a *State Space description*. These model types are currently still the most widely used and give good and fast insights in the system dynamics. The model order can be predefined, but also algorithms are available that compute the optimal model order.
In the articles [Goodwin 89] and [Goodwin 92] Goodwin et alii plead for a stochastic prior model for the distribution of unmodelled dynamics. They state that previous results on error estimation in the late 80's have relied upon prior assumptions about the noise (a known distribution or a known hard bound) and of assumed prior magnitude and smoothness bounds on the unmodelled dynamics. This was done in the form of parameterized bounding functions. The hard bounding approach leads to overly conservative error bounds. A stochastic approach of undermodelling errors is consistent with the stochastic prior description of the noise. Hereby the conservatism will be avoided. The resulting frequency-domain bounds are confidence regions.
For example, a typical *hard-bound prior assumption* on the unmodelled dynamics is that the magnitude of it's impulse response is bounded by a first-order exponential $\alpha\lambda^k$, while a typical *stochastic prior assumption* is that the variance of the unmodelled dynamics is bounded by $\alpha\lambda^k$. It will be shown that the parameters α and λ do not need to be a priori known. Only the structure of the undermodelling has to be chosen a priori and the parameters can be estimated from the data, using a maximum likelihood technique. Only the structure of a parameterized *probability density function* (PDF) has to be chosen for the prior assumptions and not the values of the parameters. The requirement on prior information is thus reduced from a quantitative one to a qualitative one. The precise form of the pdf for the undermodelling does not appear to be essential, the undermodelling only has to be stable.

The errors in the estimated transfer functions have two components: the *variance error*,

caused by the noise in the data and the *bias error*, caused by undermodelling. A classical tool for the computation of variance errors is the *Cramer-Rao lower bound.* In the case of exact model structure, which means in our case that the true process can be modelled by a FIR (see the assumptions of §2.2.3), this tool produces the smallest possible variance error expressions. However with the (practical) restricted complexity models the classical Cramer-Rao expression does not apply. The bias error is, in the case of noiseless data, a trivial problem. One can estimate as many parameters of a FIR model as there are data points N. If N is large enough, such a high-order model can be as close as possible to the true (linear) system. So if a low-order model is extracted for control design purpose, the exact bias can be computed. The case of a finite set of noisy data is more difficult. (If the noise is ZMWN, uncorrelated with the input then asymptotically the same argument applies as in the noiseless case since the noise is averaged out.)

3.1 Model estimation using Least Squares

It is assumed that the observed data are generated by the system with a FIR model structure (2.1)
One of the Prediction error methods is the *Least Squares* method. To prevent positive and negative prediction errors from cancelling out, the square of the errors is taken in the Cost Function. The *parameter vector* for the prediction of output y_k is $\Theta = [\theta_0, ..., \theta_{p-1}]$. $\Theta \in \mathbb{R}^p$, so p is the number of unknown parameters. Θ_0 is the vector of the *nominal* / true parameters for which the model describes the system exactly. The optimal solution of the parameters $\hat{\Theta}_N$ is estimated from the data $[u_k, y_k]$ via the *Classical Least-Squares*:

$$\hat{\Theta}_N = \underset{\Theta}{argmin} \frac{1}{N}\sum_{k=1}^{N} \epsilon_k^2(\Theta).$$

The prediction error $\varepsilon_k(\Theta) = y_k - \hat{y}_k(\Theta)$ is also called the *residual error.* The estimation with N data samples is denoted by the hat ^ and the subscript N. Minimization of the square of the errors is in fact minimization of the energy in the residues. This also means that in Least Squares estimation large errors get a larger weight than small errors.
The output can be rewritten as: $y_k = \hat{y}_k + \varepsilon_k = \hat{G}_N(q)u_k + \varepsilon_k$ with $\hat{G}_N(e^{-j\omega}, \hat{\Theta}_N)$ the estimated model.

The *true transfer function* $G_T(e^{-j\omega})$ is assumed to be a stochastic process. It can be decomposed as:

$$G_T(e^{-j\omega}) = G_0(e^{-j\omega}, \Theta_0) + G_{\Delta,0}(e^{-j\omega}).$$

G_T is the sum of the *nominal /prior* model G_0 and the *prior residual error* model $G_{\Delta,0}$.
$\mathcal{E}\{G_T(e^{-j\omega})\} = G_0(e^{-j\omega}, \Theta_0)$, so $G_{\Delta,0}(e^{-j\omega})$ is a zero mean stochastic process.

After model estimation, using least squares, the true transfer function can be decomposed as:

$$G_T(e^{-j\omega}) = \hat{G}_N(e^{-j\omega},\hat{\Theta}_N) + \hat{G}_{\Delta,N}(e^{-j\omega}).$$

This is the sum of the *final / estimated* model $\hat{G}_N$ and the *posteriori residual error* model $\hat{G}_{\Delta,N}$. The parameter estimation algorithm maps the a priori information $G_{\Delta,0}$ into a posteriori modelling error $\hat{G}_{\Delta,N}$. The key feature of the procedure is the quantification of the expected value of $|\hat{G}_{\Delta,N}|^2$ as a function of the properties of $G_{\Delta,0}$.

The prior residual error model $G_{\Delta,0}(q)$ is the *stochastic embedding model* with *probability density function* (PDF) $\mathbf{f}_\Delta(G_{\Delta,0}, \beta)$ where β is a real vector parametrizing the PDF. The structure of the PDF has to be specified a priori and, in the end also a value for β has to be substituted. Since $G_{\Delta,0}$ is zero mean, the value β of f_Δ affects only the second and higher order properties of $G_{\Delta,0}$. So specifying β does not amount to estimating $\hat{G}_{\Delta,N}$, but to the likely class of $\hat{G}_{\Delta,N}$. By using this class description the expected effect of the modelling error can be evaluated without the need to specify a particular realisation of $G_{\Delta,0}$. (Remark: the PDF does not need to be uniform.)

The stochastic embedding of the undermodelling $G_{\Delta,0}$ can be imposed in two ways: by specifying a prior probability distribution for the frequency function $G_{\Delta,0}(e^{-j\omega})$ or for the *impulse response sequence* $\{\eta_k\}$ of $G_{\Delta,0}(q)$.
Note that within the stochastic embedding paradigm lies the class of hard bounding solutions: specify $f_\Delta(G_{\Delta,0}, \beta)$ with compact support, i.e. at a closed and bounded interval the PDF of the stochastic embedding model $f_\Delta \neq 0$.

It is assumed that the undermodelling transfer function $G_{\Delta,0}(q)$ can be approximated sufficiently closely by a *FIR model* (a linear regression model) of order $L \leq N$. So the residual error becomes:

$$\epsilon(k) = \sum_{l=1}^{L} \eta(l)u(k-l) + e(k) = \left[\sum_{l=1}^{L} \eta(l)q^{-l}\right]u(k) + e(k),$$

where $\eta(\mathbf{k})$ is the finite impulse response of the **undermodelling** / stochastic embedding model and e(k) is ZMWN.

The transfer function is: $$G_{\Delta,0}(q) = \sum_{l=1}^{L} \eta(l)q^{-l}.$$

Let $\mathbf{f}_v(\mathbf{v}_k, \gamma)$ be the PDF of the (coloured) noise which is parameterized by the real vector γ. The form and the parameterization of this PDF have to be specified. For example take a

Gaussian distribution with expectation 0 and variance σ_v^2: $v_k \sim N(0, \sigma_v^2)$.

The system equation (2.1) can be rewritten in *regression form* linear in Θ by using Taylor's theorem:

$$y_k = \phi_k^T\Theta_0 + \psi_k^T\eta + v_k \qquad (3.1)$$

with $v_k = H(q)e_k$ unknown coloured noise uncorrelated with the input;

ϕ_k the *regression vector* with respect to the (nominal) model, a (column) vector of length p.

$\psi_k^T = [u_{k-1}, \ldots, u_{k-L}]$ a known function of the input signals, some kind of regression vector with respect to the undermodelling.

Then $\Psi = \begin{bmatrix} \psi_1 \\ \vdots \\ \psi_N \end{bmatrix} = \begin{bmatrix} u_0 & \cdots & u_{N-1} \\ \vdots & \ddots & \vdots \\ u_{1-L} & \cdots & u_{N-L} \end{bmatrix}$ a known Hankel matrix (the connection between inputs in the past and future outputs);

$\Phi^T = [\phi_1, \ldots, \phi_N]$ a known Nxp matrix; $\Phi^T\Theta$ is the *projection* of y to the columns of Φ^T;

$Y^T = [y_1, \ldots, y_N]$ the measured output;

$V^T = [v_1, \ldots, v_N]$ the form of the PDF of the noise is a priori chosen, but the parameters γ are unknown;

If the model has a FIR or ARX structure then only the transfer function G_T is estimated and the minimum of the Cost Function is the place where the gradient of the Cost Function is 0 and can be found by solving a set of linear equations, the *Normal Equations* (see for example [Stochastische Systeemtheorie]). The optimal solution $\hat{\Theta}_N$ becomes $\hat{\Theta}_N = (\Phi^T\Phi)^{-1}\Phi^TY = QY$ (3.2)

Q is a known function of the signals (time domain). This Least Squares solution is linear in Θ and the algorithm is fast. If also the noise transfer function H is estimated (for example ARMAX, output Error and Box-Jenkins models), then the Least Squares problem is no longer linear in Θ and a time-consuming iterative search process is started.

In general the true parameters of a process can never be found, but there is a bias. Substituting the system equation in regression form for Y, the optimal solution can be written as

$$\hat{\Theta}_N = \left(\frac{1}{N}\Phi^T(t)\Phi(t)\right)^{-1}\frac{1}{N}\sum_{k=1}^{N}\Phi(t)\left(\Phi^T(t)\Theta_0 + v_0(t)\right) =$$

$$\Theta_0 + \left(\frac{1}{N}\Phi^T(t)\Phi(t)\right)^{-1}\frac{1}{N}\sum_{k=1}^{N}\Phi(t)v_0(t).$$

It is clear that the parameter estimate is unbiased if the second part of the right hand side is 0. In [Stochastische Systeemtheorie] the conditions for this are given, but these conditions are never met in practical identification, because in general the system is not in the modelset. The Least Squares Estimate exists if $1/N\Phi^T\Phi$, which is $\text{cov}\{\Phi\}$ for $N \rightarrow \infty$, is invertible.

Remark: Rewrite (3.1) as $y_k = \phi_k^T\Theta_0+\varepsilon_0$. The undermodelling and noise are replaced by the true (nominal) value of the residues.
Multiplication of the expression for the nominal residues, with Q results in:

$$Q\varepsilon_0 = QY-Q\Phi\Theta_0 = \hat{\Theta}_N-\Theta_0 = \Theta, \qquad \text{because } Q\Phi = (\Phi^T\Phi)^{-1}\Phi^T\Phi = I.$$

So the difference between the estimated parameter vector and the nominal one can be expressed by the product of the nominal residues and the matrix Q, which resulted from the Least Squares. Since we are interested in this difference this is a promising notion and it will reappear later on in paragraph 3.3.

3.2 A prior PDF for the impulse response of the undermodelling

The algorithm which is based on a prior PDF for the impulse response η of the undermodelling fits properly to the environment of paragraph 3.1. This algorithm is presented in [Goodwin 92]. As described in paragraph 3.1, the structure of the probability density functions of the undermodelling $f_\Delta(G_{\Delta,0},\beta)$ and of the noise $f_\Delta(v_k,\gamma)$ have to be chosen a priori. In §3.2.2. it will be shown that, in case of a PDF for the impulse response, the parameters β can be estimated from the data. With this algorithm one can also try to estimate the noise parameters γ. This often works, but it will *not* always converge.

3.2.1 The structure of the model error

It has been assumed that the undermodelling error model $G_{\Delta,0}(q^{-1})$ can be approximated by an L-th order FIR model so the transfer function in the frequency domain is $G_{\Delta,0}(e^{j\omega}) = \sum_{l=1}^{L} \eta(l)e^{-j\omega l}$. This is in fact almost the Discrete Fourier Transform (DFT) of η. In the DFT the unit circle in the complex plane is run through exactly one time. This is in general not the case with this expression for $G_{\Delta,0}$, dependent on the values of ω and L. The stochastic embedding model can be written as $\mathbf{G}_{\Delta,0}(e^{-j\omega}) = \Pi(e^{-j\omega})\eta$,

with $\Pi(e^{-j\omega}) = [e^{-j\omega}, ..., e^{-j\omega L}]$ something like the DFT operator, a known function of ω

and $\eta^T = [\eta_1, ... ,\eta_L]$ the *finite* impulse response η_k of the stochastic embedding model.

Remark: because Π is not exactly the DFT operator, the multiplication of Π and η can *not* be replaced by the Fourier transform of η.

By introduction of Π tractable expressions for the second order properties of $G_{\Delta,0}$ are obtained.

For the nominal model it is assumed that the parametrization is a mapping to *rational transfer function* $G(q,\Theta)$ with a fixed denominator (only the numerator is parametrized by Θ) and so the transfer function is parametrized linearly in Θ.

Then the nominal model can be written as $G_0(e^{-j\omega}, \Theta_0) = \Lambda(e^{-j\omega})\Theta_0$

where $\Lambda(e^{-j\omega}) = [\Lambda_1(e^{-j\omega}), ... , \Lambda_p(e^{-j\omega})]$. Λ is a known function of ω.

The nominal model looks like the expression:

$$G_0(q,\Theta) = \frac{\theta_1+\theta_2 q^{-1}+...+\theta_p q^{-(p-1)}}{a_1+a_2 q^{-1}+...+a_r q^{-(r-1)}} \qquad (3.3)$$

and

$$\Lambda_l(e^{-j\omega}) = \frac{e^{-lj\omega}}{a_1 + a_2 e^{-j\omega} + \ldots + a_r e^{-(r-1)j\omega}} \qquad (for\ l = 0,\ldots,p-1).$$

Now the problem arises how to choose or determine the coefficients a_i of the denominator. This is the tricky part of this algorithm. In the examples of [Goodwin 92] they 'overcome' this problem by using a denominator with fixed coefficients which are determined by Laguerre interpolation. This Laguerre interpolation is known to give good low-order approximations to higher-order systems, but for this they use prior knowledge about the poles and zero's of the true system. This is a lot of extra prior knowledge. As a more practical solution, it will be suggested to use the standard ARX identification procedures and then use the resulting denominators in the error algorithm. In fact then a lot of faith is put into the quality of the denominator. (See chapter 4 on the implementation in Matlab.)

Remarks:
- if l = 0 is excluded, then there is no direct feedthrough in the model;
- in this case it is clear what p means, p-1 is the order of the numerator of the nominal model;
- here the order of the denominator (and therefore the order of the model) is equal to r; (in the Matlab 'System Identification' toolbox polynomials in q are used and the order of the numerator is required to be smaller or equal to the order of the denominator so if (3.3) is implemented in Matlab in the q domain then $p \leq r$).

The covariance of the difference between the estimated parameter vector and the nominal (true) vector is $\mathrm{cov}(\hat{\Theta}_N-\Theta_0) = \mathcal{E}\{(\hat{\Theta}_N-\Theta_0)(\hat{\Theta}_N-\Theta_0)^T\}$ (since Θ is real $\Theta^{*T} = \Theta^T$). It is straightforward how this can be written, by substituting the system equation in regression form (3.1) for Y in the optimal solution of the least squares problem (3.2).

$$(\hat{\Theta}_N-\Theta_0)(\hat{\Theta}_N-\Theta_0)^T = Q(Y-\Phi\Theta_0)(Y-\Phi\Theta_0)^TQ^T = Q(\Phi\Theta_0+\Psi\eta+V-\Phi\Theta_0)\,(\Theta_0\Phi+\eta\Psi+V-\Phi\Theta_0)^TQ^T =$$
$$= Q(\Psi\eta\eta^T\Psi^T + \Psi\eta V^T + V\eta^T\Psi^T + VV^T)Q^T.$$

When taking the estimation, then $\mathcal{E}\{\Psi\eta_k v_k\}$ is 0 since $\{\eta_k\}$ is assumed independent of $\{v_k\}$. For the covariance the following remains: $\mathrm{cov}(\hat{\Theta}_N-\Theta_0) = Q(\Psi\mathcal{E}\{\eta\eta^T\}\Psi^T+\mathcal{E}\{VV^T\})Q$.

$$\mathbf{cov}(\hat{\Theta}_N - \Theta_0) = (\Phi^T\Phi)^{-1}\Phi^T(\Psi C_\eta \Psi^T + C_v)\Phi(\Phi^T\Phi)^{-1}$$

where $C_\eta = \mathcal{E}\{\eta\eta^T\}$, the covariance of the stochastic embedding impulse response;
$C_v = \mathcal{E}\{VV^T\}$, the covariance of the noise.

Prior assumptions on the likely nature of the undermodelling can thus be translated into probable influences on $\hat{\Theta}_N$.

The goal is to calculate the covariance of the parameter vector $\mathrm{cov}\{\hat{\Theta}_N\}$ and hence the covariance of the total error $\mathrm{cov}\{G_T(q) - \hat{G}_N(q,\hat{\Theta}_N)\}$.

Ljung showed that for an impulse response of the undermodelling $\eta = 0$ under weak conditions the parameter vector $\hat{\Theta}_N \rightarrow \Theta_*$ where

$$\Theta_* = \underset{\Theta}{argmin} \frac{1}{N}\sum_{k=1}^{N} \mathcal{E}\{\epsilon_k^2(\Theta)\}.$$

The *estimation* of the squared error is taken.

Theorem 3.1a: The ***total error*** $G_T(e^{-j\omega}) - \hat{G}_N(e^{-j\omega},\hat{\Theta}_N) =$
$= G_T(e^{-j\omega}) - G(e^{-j\omega},\Theta_*) + G(e^{-j\omega},\Theta_*) - \hat{G}_N(e^{-j\omega},\hat{\Theta}_N) =$
$= (\Pi - \Lambda Q\Psi)\eta - \Lambda QV$

consists of:
- *noise / variance error* $G(e^{-j\omega},\Theta_*) - \hat{G}_N(e^{-j\omega},\hat{\Theta}_N) = \Lambda QV$;
- a *bias / undermodelling error* $G_T(e^{-j\omega}) - G(e^{-j\omega},\Theta_*) = (\Pi - \Lambda Q\Psi)\eta$.

The noise error is a random variable (vanishes when there is no noise or when the number of data tends to infinity). In classical identification theory the undermodelling error is a deterministic quantity, but now, with the stochastic embedding model it becomes also a random variable. In this undermodelling error:

$\Pi\eta$	$= G_{\Delta,0}$, the prior estimate of the undermodelling; the true undermodelling;
$\Lambda Q\Psi\eta$	is a data-induced error (and not a correction like Goodwin e.a. write) to the prior estimate due to the shift from Θ_0 to $\hat{\Theta}_N$ which arises in the Least Squares procedure.

The model error is a linear combination of two independent random vectors η and V.

Proof: $G_T(e^{-j\omega}) = \Lambda\Theta_0+\Pi\eta$ and
$\hat{G}_N(e^{-j\omega},\hat{\Theta}_N) = \Lambda\hat{\Theta}_N$ so
$G_T(e^{-j\omega}) - \hat{G}_N(e^{-j\omega},\hat{\Theta}_N) = \Lambda(\Theta_0-\hat{\Theta}_N) + \Pi\eta$.
Using the optimal solution and substituting the system equation for Y gives
$\Theta_0-\hat{\Theta}_N = \Theta_0-QY = \Theta_0-Q(\Phi\Theta_0+\Psi\eta+V) = -Q\Psi\eta-QV$ (remark: $-Q\Phi\Theta_0 = -\Theta_0$).
Finally substituting this in the error equation gives the theorem.
□

Theorem 3.1b: The ***mean square error*** follows by the definition of the covariance:

$$\mathcal{E}\{|\hat{G}_N(e^{-j\omega},\hat{\Theta}_N) - G_T(e^{-j\omega})|^2\} = (\Pi - \Lambda Q\Psi)C_\eta(\Pi - \Lambda Q\Psi)^{T^*} + \Lambda QC_vQ^T\Lambda^{T^*} =$$
$$= \Pi C_\eta\Pi^{T^*} - \Pi C_\eta(\Lambda Q\Psi)^{T^*} - \Lambda Q\Psi C_\eta\Pi^{T^*} + \Lambda Q\Psi C_\eta(\Lambda Q\Psi)^{T^*} + \Lambda QC_vQ^T\Lambda^{T^*}.$$

All the quantities are known except for the covariances C_η and C_v which are a priori chosen functions of the unknown parameter vectors β and γ. If these parameters are available or can

be estimated from the data, then computable estimates of the mean square error of the estimated transfer functions are obtained.
Note: Π^* is something like the inverse DFT operation.

3.2.2 Estimation of the parameters of the noise and undermodelling

The residuals ε perform a projection of y on the orthoplement P of Φ. With Q from the Least Squares algorithm the residuals can be written as:

$$\varepsilon = Y-\hat{Y} = Y - \Phi\hat{\Theta}_N = [I - \Phi(\Phi^T\Phi)^{-1}\Phi^T]Y = PY.$$

Since the total space has dimension N and the parameter vector Θ has dimension p, the regression vector Φ has at most rank p. So the orthoplement P has at least rank N-p. The residuals are overdetermined and so ε has an (unwanted) singular distribution. To obtain a new *full rank data vector* W, ε is represented in a new coordinate system. This system forms a basis for the space orthogonal to the columns of Φ. Let R be any matrix whose columns span the subspace orthogonal to the columns of Φ. One way of constructing R is to take any N-p independent linear combination of the columns of P.
Then $W = R^T\varepsilon$ and by writing the residuals ε with the expression derived above

$$\mathbf{W = R^T(Y-\hat{Y}) = R^T[I - \Phi(\Phi^T\Phi)^{-1}\Phi^T]Y = R^TY} \qquad (3.4).$$

This is because R is orthogonal to Φ.
Writing ε with the system equation in regression form $W = R^T(Y-\hat{Y}) = R^T(\Phi\Theta_0+\Psi\eta+V-\Phi\Theta_0)$ $= R^T\Psi\eta + R^TV$. So W is the sum of two independent random vectors η and V whose probability functions are computable functions of the unknown parameters. W is a function of the input signal u (R^T and Ψ depend on the input signal only) and the unknown covariance parameters $\xi = [\beta, \gamma]$. W has a *non*singular distribution. $W \in \mathbb{R}^{N-p}$ and depends only on $\eta_{p+1},\ldots,\eta_N$, i.e. that part of the infinite impulse response that is included in the nominal model is eliminated from the data W.
The corresponding likelihood function is denoted by $\mathfrak{L}(W \mid U, \xi)$,

which has to be maximized for ξ: $$\hat{\xi} = \underset{\xi}{argmax}\{\mathfrak{L}(W|U,\xi)\}.$$

Maximizing this likelihood function yields indeed the desired estimate for the unknown parameters β of the stochastic embedding model, but unfortunately it is *not* guaranteed that the parameters γ of the noise can be estimated.
The parameters ξ can then be substituted in theorem 3.1b of §3.2.1 and the estimation of the model error results.

3.2.3 Case of a Gaussian embedding

Assume that the impulse response η of the stochastic embedding model (dimension L) has a Gaussian / Normal distribution $\eta \sim N(\mu, \sigma_\eta^2)$:

$$\frac{1}{\sigma_\eta\sqrt{2\pi}}\, e^{-\frac{(\eta-\mu)^2}{2\sigma_\eta^2}}$$

with expectation $\mu = 0$ because $\mathcal{E}\{G_{\Delta,0}\} = 0$ and standard deviation σ_η (variance σ_η^2).

Assume that the variance of the undermodelling is bounded by a first order exponetial, so the covariance (LxL) matrix becomes $C_\eta = \text{diag}\{\alpha\lambda^k\}$ $(1 \le k \le L)$,
α and λ are unknown. So $\eta \sim N(0, C_\eta(\beta))$.
Use also a zero mean Gaussian distribution for the output noise: $v_k \sim N(0,\sigma_v^2)$ (σ_v^2 is unknown).
The covariance of the noise is $C_v = \sigma_v^2 I$, a NxN matrix.
The input noise is independent of η.
Without loss of generality the order L of the FIR model can be taken equal to the number of data samples N (**N=L**). In this case regression vector Φ contains shifted versions of the input:

$$\Phi = \begin{bmatrix} u_0 & \cdots & u_{1-p} \\ \vdots & & \vdots \\ u_{N-1} & \cdots & u_0 \end{bmatrix} \quad \text{and} \quad \Psi = \begin{bmatrix} u_0 & \cdots & u_{1-N} \\ \vdots & & \vdots \\ u_{N-1} & \cdots & u_0 \end{bmatrix}.$$

So in this case the parameters of the PDF of the undermodelling error model are $\beta^T = [\alpha,\lambda]$ and the vector of all unknown parameters is $\xi^T = [\beta, \sigma_v^2]$. The matrix R, whose columns span the subspace orthogonal to the columns of Φ, can be constructed by the last N-p columns of

Ψ: $$R = \begin{bmatrix} u_{1-p-1} & \cdots & u_{1-N} \\ \vdots & & \vdots \\ u_{N-p-1} & \cdots & u_0 \end{bmatrix}$$

and the new data vector W (with a non-singular distribution) is: $W = R^T Y$.
The Gaussian assumptions on the distributions f_Δ and f_v give a *log likelihood function* for the observed data:

$$\ell(W \mid U, \xi) = -\tfrac{1}{2}\ln(\det(\Sigma)) - \tfrac{1}{2}W^T\Sigma^{-1}W + \text{constant} \qquad (3.5)$$

where $\Sigma = R^T\Psi C_\eta \Psi^T R + \sigma_v^2 R^T R$.
So the unknown parameters can be found from

$$\hat{\xi} = \underset{\xi}{\textit{argmax}}\ \{\ell(W \mid U, \xi)\}.$$

3.2.4 Cramer-Rao lower bounds

The covariance of an *unbiased estimate* $\hat{\xi}$ of ξ is bounded below by the Cramer-Rao lower bound: $\mathrm{cov}\{\xi\} \geq M_\xi^{-1} \triangleq \underline{\mathrm{cov}}\{\xi\}$.
Where M_ξ^{-1} is the *Fisher information matrix*. $\underline{\mathrm{cov}}$ is often a good guide to the covariance of estimators which are not actually unbiased. For the case of Gaussian assumptions, an explicit expression for the information matrix can be computed:

$$M_\xi = \frac{1}{2}\begin{bmatrix} tr\{(\Sigma^{-1}T)^2\} & tr\{\Sigma^{-1}T\Sigma^{-1}K\} & tr\{\Sigma^{-1}T\Sigma^{-1}\Delta\} \\ tr\{\Sigma^{-1}T\Sigma^{-1}K\} & tr\{(\Sigma^{-1}K)^2\} & tr\{\Sigma^{-1}K\Sigma^{-1}\Delta\} \\ tr\{\Sigma^{-1}T\Sigma^{-1}\Delta\} & tr\{\Sigma^{-1}K\Sigma^{-1}\Delta\} & tr\{(\Sigma^{-1}\Delta)^2\} \end{bmatrix}$$

where $$T = R^T\Psi\Upsilon\Psi^T R = \frac{\partial\Sigma}{\partial\alpha}$$

$$K = R^T\Psi\Xi\Psi^T R = \frac{\partial\Sigma}{\partial\lambda} \qquad \Delta = R^T R = \frac{\partial\Sigma}{\partial\sigma_v^2}$$

$$\textit{with } \Upsilon = diag(\lambda, \lambda^2, ..., \lambda^L) = \frac{\partial C_\eta}{\partial\alpha} \quad \textit{and} \quad \Xi = diag(\alpha, 2\alpha\lambda, ..., L\alpha\lambda^{L-1}) = \frac{\partial\Sigma}{\partial\lambda}.$$

See [Goodwin 92] for the proof of these expressions.
The availability of M_ξ and the simple form of the likelihood function ℓ in the case of Gaussian embedding is a motivation to use the Gaussian assumption.

In case of the situation as described above the unbiased estimate $\hat{\xi}$ of ξ has the following asymptotic properties:

$$\underline{cov}\left(\frac{\bar{\alpha}}{\alpha}\right) = O\left(\frac{1}{\ln N}\right), \qquad \underline{cov}\left(\frac{\bar{\lambda}}{\lambda}\right) = O\left(\frac{1}{(\ln N)^3}\right) \quad \textit{and} \quad \underline{cov}\left(\frac{\bar{\sigma}_v^2}{\sigma_0^2}\right) = O\left(\frac{1}{N}\right).$$

The maximization problem is convex. The estimate $\bar{\sigma}_v^2$, is asymptotically decoupled from the estimates $\bar{\alpha}$ and $\bar{\lambda}$. The proof of this can be found in [Goodwin 92]. This is significant since it shows that the variance of the estimates decays with increasing data length and therefore it can be expected that the estimates converge to their true values.

3.3 A prior PDF for the frequency function of the undermodelling

Here the algorithm according to [Goodwin 89] is presented. This algorithm is in the continuous time and frequency domain. This has the advantage that analytical solutions can be computed for the most simple cases. Unfortunately the implementation in Matlab will be more difficult (see chapter 4). Only the undermodelling error is regarded (no noise). Not only the structure of the Probability Density Function of the undermodelling frequency function is specified a priori, but also the parameters of the error function are specified prior to the identification experiment. As a result, the embedding of this paragraph is a little bit different from the description of §3.1.

3.3.1 A nominal ARX model

$G_T(s) = G_0(s) + G_\Delta(s)$, in this case $G_\Delta(s)$ only contains the undermodelling error.
For the nominal model an ARX structure is assumed, of which the block scheme can be seen in figure 3.2. However in this setting noise is left out so then remains:

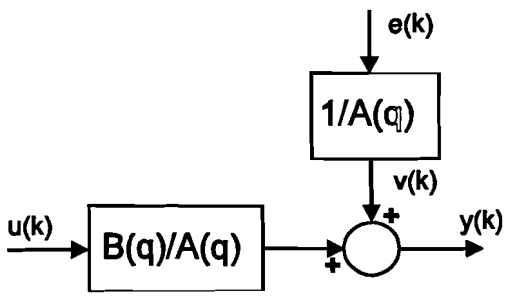

figure 3.2: ARX model structure

$G(\Theta, j\omega) = B(\Theta, s)/A(\Theta, s)$ with

$B(\Theta, s) = b_m s^m + b_{m-1} s^{m-1} + \ldots + b_0$;

$A(\Theta, s) = s^n + a_{n-1} s^{n-1} + \ldots + a_0$.

$\Theta^T = [b_m , b_{m-1} , \ldots, b_0 , a_{n-1} , \ldots, a_0]$, the parameter vector.

The residuals ε, represent the undermodelling error, related to u(t) by $G_{\Delta,0}$. A *linear* regression form of the system equation is obtained by using Taylor's theorem: $y(t) = \phi(t)^T\Theta_0 + \varepsilon(t)$.
Now the regression vector is $\phi(t)^T = [u^{(m)}(t), \ldots, u(t), y^{(n-1)}(t), \ldots, y(t)]$, where $^{(m)}$ denotes the m-th derivative. m and n are the orders of the B and A polynomials respectively of the chosen nominal model.
The solution is (see also (3.2)):

$$\hat{\Theta}_N = QY = \left(\frac{1}{T}\int_0^T \phi(t)\phi^T(t)dt\right)^{-1} \frac{1}{T}\int_0^T \phi(t)y(t)dt = P\frac{1}{T}\int_0^T \phi(t)y(t)dt$$

with

$$P = \left(\frac{1}{T}\int_0^T \phi(t)\phi^T(t)dt\right)^{-1}.$$

(Note that the factor 1/T is included in the inverse operation.)
Then the difference between the final parameter vector and the nominal estimations of the parameters becomes:

$$\tilde{\Theta} = \hat{\Theta}_N - \Theta_0 = P\frac{1}{T}\int_0^T \phi(t)\eta(t)dt.$$

Again $\eta(t)$ is the impulse response of the undermodelling. This is in agreement with the remark at the end of paragraph 3.1 because here $\eta(t) = \varepsilon(t)$.
The stochastic embedding of the undermodelling model G_Δ is introduced by a PDF for the frequency function of G_Δ in the form of a parametrized covariance function $\text{cov}\{G_\Delta(\omega_1, \omega_2)\} = \mathcal{E}\{|G_\Delta|^2\}$.

Theorem 3.2: The difference between the true system transfer function and the optimal/final estimated one, satisfies:

$$\mathcal{E}\{|\hat{G}_{\Delta,N}|^2\} = \mathcal{E}\{|G_{\Delta,0}|^2\} + \mathcal{E}\{|\hat{G}_N - G_0|^2\} - 2\text{Re}\mathcal{E}\{(\hat{G}_N - G_0)G_{\Delta,0}^*\}$$

with $\mathcal{E}\{|G_{\Delta,0}|^2\} = \text{cov}\{G_{\Delta,0}(\omega_1, \omega_2)\}$, the prior estimate of the undermodelling error

and

$$\hat{G}_N(j\omega) - G_0(j\omega) \approx \frac{\hat{A}_N(\hat{\Theta}_N, s)}{A_0(\Theta_0, s)}\Big|_{s=j\omega} \frac{\partial G(j\omega)^T}{\partial\Theta}\Big|_{\Theta=\hat{\Theta}_N} (\hat{\Theta}_N - \Theta_0)$$

Proof: A real proof of theorem 3.2 will not be given here. For the proof the reader is referred to [Goodwin 89]. Here only a short elucidation is given.
The nominal model can be approximated by a first order Taylor approximation of the estimated model:

$$G_0 = \frac{B_0(\Theta_0,s)}{A_0(\Theta_0,s)} = \hat{G}_N + G_\Delta' \approx \hat{B}_N(\hat{\Theta}_N,s) + \frac{\hat{A}_N(\hat{\Theta}_N,s)}{A_0(\Theta_0,s)}\Big|_{s=j\omega} \frac{\partial G}{\partial\Theta}\Big|_{\Theta=\hat{\Theta}_N} (\Theta_0 - \hat{\Theta}_N) \Leftrightarrow$$

$$G_\Delta' = G_0 - \hat{G}_N \approx \frac{\hat{A}_N(\hat{\Theta}_N,s)}{A_0(\Theta_0,s)}\Big|_{s=j\omega} \frac{\partial G}{\partial\Theta}\Big|_{\Theta=\hat{\Theta}_N} (\Theta_0 - \hat{\Theta}_N)$$

If the term $\hat{A}_N/A_0$ in some way becomes equal to 1 (the estimated coefficients of A(s) are equal to the nominal ones), then the model, which originally had an ARX structure, becomes parametrized linearly in Θ.

□

3.3.2 $G_{\Delta,0}$ a wide-sense stationary process and ϕ a function of only exogenous variables

If a nominal FIR model without output noise is used (see figure 3.3), then the regression vector is only a function of the input. In the theorem 3.2 the following simplifications can be used:

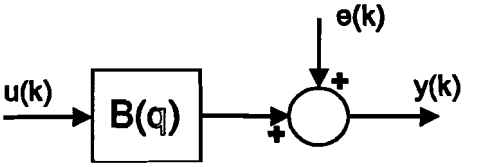

figure 3.3: FIR model structure

- the term $\hat{A}_N/A_0$ is equal to unity and so the result is not a function of the unknown nominal model;
- the error $\varepsilon(t)$ is a linear function of the system input and of the unmodelled dynamics and hence not a function of the unknown nominal model;
- $\left.\frac{\partial G^T}{\partial \Theta}\right|_{\Theta=\hat{\Theta}_N} = P^m(j\omega)$ where $P^m(\rho_t)$ has been defined in §2.1 as the vector of derivative operators ρ_t in decreasing order in time domain and $P^m(j\omega)$ is the transformation to the frequency domain; it is a known function of frequency.

If the frequency domain process $G_{\Delta,0}$ is a *wide-sense stationary process*, then the covariance function $\mathrm{cov}\{G_{\Delta,0}(\omega_1,\omega_2)\}$ will depend only on the difference in frequency $\omega_1-\omega_2$. For this function $\mathrm{cov}\{G_{\Delta,0}(\omega_1,\omega_2)\}$ it holds: $cov\{G_{\Delta,0}(\omega_1, \omega_2)\} = \overline{cov}\{G_{\Delta,0}(\omega_1-\omega_2)\}$.

Define

$$\overline{cov}G_{\Delta,0}(\omega_1-\omega_2) = \frac{\sigma_0^2}{j(\omega_1-\omega_2)+\beta} = \sigma_0^2\frac{\beta}{(\omega_1-\omega_2)^2+\beta^2} - \sigma_0^2\frac{j(\omega_1-\omega_2)}{(\omega_1-\omega_2)^2+\beta^2} \quad (\beta>0) \qquad (3.6).$$

The real part is an even function and the imaginary part an odd function of the difference $\Delta\omega$, as can be seen in figure 3.4 for $\sigma_0^2 = \beta = 10$.

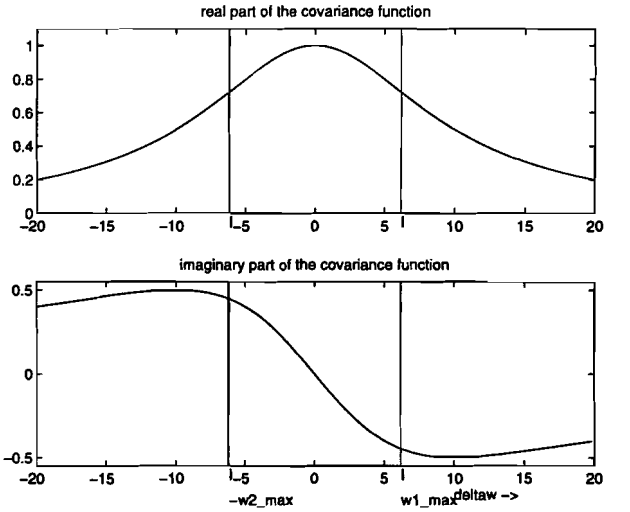

figure 3.4: The real and imaginary part of the covariance function

For the corresponding time domain impulse response η of the undermodelling $\mathrm{cov}\{\eta\} = \sigma_0^2 e^{-\beta t}$.

So then in theorem 3.2

$$cov\{G_{\Delta,0}(\omega,\omega)\} = \frac{\sigma_0^2}{\beta}$$ which is a dc-component.

For the numerical implementation an additional filter D(s) will be used such that the regression vector can be computed with integration operations. The numerical algorithms for integrations are better than those for derivatives in which the noise is amplified. (See chapter 4.) In time domain D is a function of the derivative operator ρ_t and is denoted as $d(\rho_t)$.

The regression vector becomes:

$$\phi(t) = \frac{P^m(\rho_t)}{d(\rho_t)}u(t) = \begin{bmatrix} \frac{u^{(m)}(t)}{d(\rho_t)} \\ \vdots \\ \frac{u(t)}{d(\rho_t)} \end{bmatrix}.$$

In theorem 3.2:

$$\hat{G}_N - G_0 = \frac{P^m(j\omega)}{D(j\omega)}\tilde{\Theta} = \frac{P^m(j\omega)}{D(j\omega)}P\frac{1}{T}\int_0^T \phi(t)\eta(t)dt.$$

$$\mathcal{E}\{|\hat{G}_N - G_0|^2\} = \frac{P^m(j\omega)}{D(j\omega)}P\frac{1}{T^2}\int_0^T\int_0^T \phi(\tau_1)\phi(\tau_2)^T\eta(\tau_1)\eta(\tau_2)^T d\tau_1 d\tau_2 P\frac{P^m(j\omega)}{D(j\omega)}$$

Now an expression for the impulse response $\eta(t)$ of the undermodelling (the residues $\varepsilon(t)$) has to be found. The proof of this can be found in [Goodwin 89].

$$q(\tau_1,\tau_2) = \epsilon(\tau_1)\epsilon(\tau_2)^T = \frac{1}{2\pi}\int_{-\infty}^{\infty} M_1(\omega_1,\tau_2)U(\omega_1)^* e^{-j\omega_1\tau_1}d\omega_1$$, the Fourier transform

of $1/2\pi \bullet M_1(\omega_1,\tau_2)U(\omega_1)^*$ with

$$M_1(\omega_1,\tau_2) = \frac{1}{2\pi}\int_{-\infty}^{\infty} cov\{G_{\Delta,0}(\omega_2,\omega_1)\}U(\omega_2)e^{j\omega_2\tau_2}d\omega_2$$

which is the inverse Fourier transform of the multiplication of the prior estimate of the undermodelling and the Fourier transform of input u.

With the same reasoning the third term of theorem 3.2 becomes:

$$\mathcal{E}\{(\hat{G}_N - G_0)G^*_{\Delta,0}\} = \left[\frac{P^m(\omega)}{D(j\omega)}\right]^T \frac{1}{T}\int_0^T \phi(\tau_2)M_2(\omega,\tau_2)d\tau_2 \quad \text{with}$$

$$M_2(\omega,\tau_2) = \frac{1}{2\pi}\int_{-\infty}^{\infty} cov\{G_{\Delta,0}(\omega_2,\omega)\}U(\omega_2)e^{j\omega_2\tau_2}d\omega_2 .$$

3.3.3 Extension to regression vectors which depend on the output

In the more general case (nominal ARX model), when the regression vector depends upon the output, exact evaluation of $\mathcal{E}\{|\hat{G}_N - G_0|^2\}$ is not possible due to the auto-regressive parameters. Now the actual estimated parameters are necessary. A reasonable approximation can be obtained by first performing an actual process identification, using the standard procedures, and then using these results $\{\hat{\Theta}^p, \hat{A}_N^p, \hat{B}_N^p, \phi^p(t)\}$ in the error algorithm.
A filter F(s) of order n is introduced to specify the frequency region where the algorithm has to concentrate on. In [Stochastische Systeemtheorie] it is shown that when estimating ARX models, in general the fit will be better for high frequencies than for low frequencies. This is usually unwanted because the most interesting region is in general at the low frequencies. To improve the estimate for low frequencies, Low Pass Filters can be used (at the cost of lower accuracy for higher frequencies).

Then for $\mathcal{E}\{|\hat{G}_N - G_0|^2\}$ the following approximations can be used:

- the random variable $\hat{A}_N/A_0$ is again replaced by unity;
- the *exact transfer function* $D(\Theta_0,j\omega)G_{\Delta,0}(j\omega)$ linking the undermodelling error $\eta(t)$ and the input u(t) is approximated by $\dfrac{\hat{A}_N^p(j\omega)}{F(j\omega)}G_{\Delta,0}(j\omega)$

  (the orders of $\hat{A}^p$ and F are the same, equal to n);

- the random variable $\left.\dfrac{\partial G^T}{\partial\Theta}\right|_{\Theta=\hat{\Theta}}$ is replaced by the deterministic (numerical) value $\left.\dfrac{\partial G^T}{\partial\Theta}\right|_{\Theta=\hat{\Theta}^p}$, which has been determined in the identification experiment;

  $\Theta = [b_m, ..., b_0, a_{n-1}, ..., a_0]$ with length m+n+1;
- the random variable $\phi(t)$ is replaced by the measured vector $\phi^p(t)$ when evaluating P and $\hat{\Theta}_N - \Theta_0$.

These approximations are reasonable since they retain the essential linear dependence of

$\mathcal{E}\{|\hat{G}_N - G_0|^2\}$ on $G_{\Delta,0}(\omega)$ whilst eliminating higher-order effects which arise from the difference between $\phi^p(t)$ and $\phi(t)$ and between $\hat{A}_N$, A_0 and $\hat{A}_N^p$.

With these approximations the final error $\mathcal{E}\{|\hat{G}_{\Delta,N}|^2\}$ can be evaluated as in §3.3.1 save that the following terms are replaced:

- $P^m(j\omega)$ by

$$\left.\frac{\partial G}{\partial \Theta}\right|_{\Theta=\hat{\Theta}^p}^{T} = \frac{\partial}{\partial \hat{\Theta}^p}\left\{\frac{\hat{B}_N^p}{\hat{A}_N^p}\right\} = \frac{\partial}{\partial \hat{\Theta}_N^p}\left\{\frac{\hat{b}_m s^m + \hat{b}_{m-1}s^{m-1} + \ldots + \hat{b}_0}{s^n + \hat{a}_{n-1}s^{n-1} + \ldots + \hat{a}_0}\right\}$$

$$= \begin{bmatrix} \frac{s^m}{s^n + \hat{a}_{n-1}s^{n-1} + \ldots + \hat{a}_0} \\ \frac{s^{m-1}}{s^n + \hat{a}_{n-1}s^{n-1} + \ldots + \hat{a}_0} \\ \vdots \\ \frac{1}{s^n + \hat{a}_{n-1}s^{n-1} + \ldots + \hat{a}_0} \\ \frac{-(\hat{b}_m s^m + \hat{b}_{m-1}s^{m-1} + \ldots + \hat{b}_0)s^{n-1}}{(s^n + \hat{a}_{n-1}s^{n-1} + \ldots + \hat{a}_0)^2} \\ \frac{-(\hat{b}_m s^m + \hat{b}_{m-1}s^{m-1} + \ldots + \hat{b}_0)s^{n-2}}{(s^n + \hat{a}_{n-1}s^{n-1} + \ldots + \hat{a}_0)^2} \\ \vdots \\ \frac{-(\hat{b}_m s^m + \hat{b}_{m-1}s^{m-1} + \ldots + \hat{b}_0)1}{(s^n + \hat{a}_{n-1}s^{n-1} + \ldots + \hat{a}_0)^2} \end{bmatrix} \qquad (3.7)$$

The quotient rule is used. This derivative is a column vector of length m+1+n.

- replace U(.) by a filtered version $U_f = \frac{\hat{A}_N^p(.)}{F(.)}U(.)$

and u(.) by $u_f = \frac{\hat{A}_N^p(.)}{f(.)}u(.)$;

- and replace $\phi(t)$ by $\phi^p(t) = \frac{P(\rho_t)}{d(\rho_t)} \begin{bmatrix} \frac{\hat{A}_N^p(t)}{f(t)} u(t) \\ y(t) \end{bmatrix}$.

3.3.4 Extension to non-stationary frequency domain description

There are situations where a *non*-stationary frequency domain description of the modelling error would be more appropriate than the simple stationary description. The main problem with the stationary description is that the variance of the modelling error is independent of frequency. However, it is usually the case that at high frequencies the true system response decays to zero and thus the absolute modelling error should also decay. Another situation where non-uniform frequency domain variances could be used effectively is when there are high-frequency resonant modes which are not included in the nominal model. If the frequency bands are known in which these resonances occur, then it is possible to allocate the undermodelling variance accordingly.

There are many different ways in which a non-stationary covariance function can be specified. One way is to choose $\text{cov}\{G_{\Delta,0}(\omega_1,\omega_2)\} = S(\omega_1-\omega_2)N(\omega_1+\omega_2)$, where it is required that the stationary factor S satisfies $S(\omega) = S(-\omega)^*$ and the non-stationary factor N is a real even function of ω to ensure that the corresponding time domain functions are real.

Remark: the stationary part in the frequency domain maps into the non-stationary part in the time domain and vice versa (see [Goodwin 89]).

The particular covariance function which satisfies this conditions and which has been used in the Matlab implementation is with:

$$N(\omega) = \frac{4\beta K_1}{\omega^2+4\beta^2}$$
$$S(\omega) = \frac{K_2}{j\omega-\beta} \qquad (3.8).$$

The real and imaginary part of the non-stationary covariance function are plotted in figure 3.5 for $\beta = 2$, $K_1 = 20$ and $K_2 = 8$.

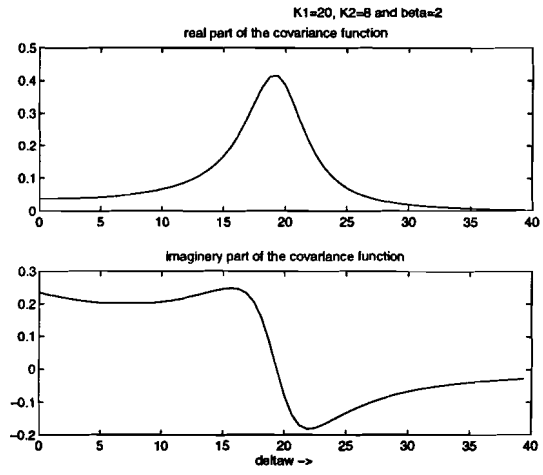

figure 3.5: The real and imaginary part of the non-stationary covariance function

Remark: the use of non-stationary covariances allows for smoothness of both time domain impulse response and frequency domain smoothness (see [Goodwin 89]). S describes the envelope of the impulse response and N the smoothness of the impulse response. This is helpful in given intuitive guidance when choosing the parameters of the covariance.

4. Implementation in Matlab

4.1 The structure of the implementation of the stochastic approach

The two algorithms of Goodwin which have been treated in the last chapter are combined in one implementation in Matlab. The main programme file is *goodwin.m* and this m-file can be called in Matlab.

4.1.1 The subroutines

The following subroutines (with a short description) are used:

GENERAL FILES:

sys_and_.m Definition of the system and the signals which will be used in case of an internal simulation.

display.m Returns the values of parameters in the Matlab figure window after the default values of the parameters are changed by the user. This changing is done by sliders.

model_es.m Estimates an ARX model of predefined order and delay with the standard methods of the 'Identification Toolbox' and performs a simulation of the estimated model with the input currently in use.

plot_goo.m Takes care of the plotting of the signals and quantities that have been computed.

val.m Is called in *plot_good* and stores the chosen figures that have to be plotted.

ALGORITHM FILES:

goodwin1.m A second part of the initialisation for the algorithm with a prior assumption on the undermodelling frequency function (continuous domain, see §3.3).

goodwn1b.m The actual calculation of the undermodelling error.

cova.m Computation of the covariance function of the undermodelling frequency function. In this algorithm special care is taken to bound the covariance to a frequency interval $0 < \omega \leq 2\pi$ in order to make it usable in IFFT operations.

regres1.m Computes the regression vector which is used in the Least Squares algorithm. In this calculation a filter D(s) is included to make the algorithm numerically better, as explained in §3.3.2.

goodwin2.m Computation of the model error (both undermodelling and noise error) with prior assumption on the impulse response of the undermodelling error (see §3.2).

optimiza.m The unknown variance parameters of the undermodelling and the noise are estimated through maximization of a likelihood function.

likeli.m The log likelihood function (3.5) which has to be maximized to estimate the covariance parameters.

goodwn2b.m The following part of *goodwin2.m*, after the variance parameters have been estimated the expectation of the model error can be computed according to theorem 3.1b. This is done in a separate routine in order to make the use more flexible, especially during implementation. Otherwise each time the time-consuming optimization algorithm should have to be performed.

cramer_r.m Computes lowerbounds for the variance parameters according to Cramer-Rao (see §3.2.4).

DATA FILE

data.mat Contains the input and output data (two columns) which will be loaded in case the algorithm is used with external data.

4.1.2 An internal simulation

After the user in the Matlab prompt has typed 'goodwin' (and a <enter>) a figure window appears which describes briefly the goal of the algorithm. It is indicated that there are two approaches and a reference is given to the articles [Goodwin 89] and [Goodwin 92]. With two push-buttons the user can choose to perform an internal simulation or to use the algorithm for external data. After the user has made the choice, the Matlab prompt appears again and the user is asked to push <enter> to continue.

In case of a simulation, the true system has to be defined. As default a system is available which looks like the one of paragraph V of [Goodwin 92]. The user can choose for an other system of which the transfer function has to be defined in the Matlab prompt. After this, the number of data samples and the sample frequency have to be specified. This can be done with sliders in a figure window. The number of data points N is a power of 2 because several times Fourier- and inverse Fourier operations are performed and with N a power of 2 this can be done by the Fast Fourier Transform (FFT) algorithm. The default value is N = 64. The default value of the sample frequency f_s is 2.5 times the frequency of the fastest pole. In order to continue a pushbutton has to be used. An input signal with 0 mean has to be defined and there are 3 choices: a square wave, a sinusoidal and ZMWN. After the user has made the choice by a pushbutton, the parameters of the signals have to be specified. In case of a square wave or a sinusoidal the signal frequency and amplitude are determined, again by a slider. For the signal frequency f_0 two sliders are available, one with a range from 0 to 10 Hz and one from 0 to 1 Hz which allows to specify low frequencies more accurately. Also the variance of the output noise which will be added, has to be entered in. In case of a ZMWN

for input, the variances of this noise and of the output noise have to specified. Next the user chooses the type of model which had to be estimated with a push button: a FIR or an ARX model. In the Matlab prompt the orders of the B- and A polynomials and of the delay of the model are put in. If an ARX model is estimated, then a (Low Pass) filter 1/F(s) has to be defined to improve the estimation in the frequency region of interest. The Bode plot of the filter is shown and the user can accept or reject the filter and define a new one. The input and output are filtered with this filter. After this initialisation, the nominal model is computed. The user is asked to wait...
In a new figure window the user can choose to use the algorithm GOODWIN1 (a PDF for the frequency function of the undermodelling, noise is left out and deals only in a good way with FIR models) or GOODWIN2 (a PDF for the impulse response of the undermodelling, noise is included and has been designed to deal with both FIR and ARX).

GOODWIN1: a PDF for the frequency function of the undermodelling
In case of GOODWIN1 the PDF of the undermodelling has to be specified. One can choose the stationary description (3.6) or the non-stationary covariance function (3.8). Then the parameters of the covariance function have to be entered. The real and imaginary parts of the covariance are shown in a plot and the user can accept this covariance or use other values for the parameters. It is told that a filter 1/D(s) will be used in the calculation of the regression vector and the polynomial D(s) is shown. If an ARX model has been estimated, then the regression vector also contains versions of the output and the input is filtered with $A_0(s)/F(s)$ (see §3.3.3).

The model error is computed and the user is asked to wait...
When the algorithm is finished, a new figure window appears. Here the user can indicate which plots have to be shown:
1. A plot of the input, the estimated output and, in case of a simulation, the true output.
2. The frequency responses of both the true system (in case of a simulation) and the estimated model.
3. A plot of the frequency function of the estimation of the model error.
4. In case of GOODWIN1 the different parts of the undermodelling error are plotted separately and in case of GOODWIN2, the undermodelling error and the noise error are plotted separately.
5. Nyquist plots of the estimated model with error bars indicating the computed estimation of the model error and ,in case of simulation, also of the true system.

GOODWIN2: a PDF for the impulse response of the undermodelling
The parameters of the noise and undermodelling covariances are estimated through a time-consuming optimalization technique. However, if these parameters are available in one way or the other, the user can indicate this and then the algorithm will be much faster. The values of the parameters can be specified in the Matlab prompt. If the covariance parameters are not available, then the user has to specify the initial values of the parameters which will be used

in the first step of the optimization. This can be done by the sliders in a figure window. A good choice of the initial parameters speeds up the optimization. Finally the user is asked if the Cramer-Rao bounds have to be computed (push button).
After the initialisation the same procedure follows as with GOODWIN1.

4.1.3 External data

In case the error algorithm has to be used with external data, this choice can be made in the first figure window which appears after *goodwin* has been called in the Matlab prompt. The input and corresponding output data are read automatically from a file *data.mat*. The length of these vectors should be at least 4 times the number of data N for which the error algorithm has to be performed. A few parameters for the algorithm have to be entered: the frequency f_0 of the input signal (in case of ZMWN f_0 remains empty), the duration T of the experiment, maximal frequency f_{max} for which the algorithm will be performed and the number of samples N (preferably N is a power of 2 because then (I)FFT algorithms can be used).
After this has been done, it is asked if a model already has been estimated. If this is the case then the transfer function has to be specified. Else the type of model (FIR or ARX) has to be chosen and the polynomial orders, the delay and the low pass filter 1/F(s) have to specified. Then the routines go on like in case of an internal simulation.

4.1.4 Construction of the time and frequency vectors

There are several things to take into account when time and frequency vectors and related signals are constructed in the discrete implementation. Of course Shannon's theorem dominates the precautions which are taken to get sensible results. If an internal simulation is performed in the error routines then the sample frequency is based on the highest system frequency. If external data is used, it is assumed that the frequencies present, are bounded to prevent aliasing. Signal vectors are created which are longer than the actual number of samples N for which the error algorithm has to be performed. In calculation of the model error only the last N samples are used to eliminate all kinds of initial conditions of systems and filters. The sample period (time spacing t_δ) has to be in such a way to prevent aliasing and the length T of the time interval has to be in such way to capture all dynamics in time domain. The combination of these two factors determines the number of samples. However the complexness of the algorithms and the limitation of today's pc's restricts the number of samples for which the algorithm can be performed, in a reasonable time, to about 256. This is often a problem in practical systems which have (very) fast and (very) slow dynamics. Then the choice of the different parameters is a trial and error procedure. Some general guidelines for experiment design and data conditions are given in appendix C.

4.2 Implementation of the algorithm with a PDF for the impulse response

The Matlab implementation is of course discrete, so in the implementation the representation of a scalar function is a vector of samples in time or frequency domain. To make a difference between one sample and the whole vector, the vector will be underlined. The time vector $\underline{t}$ and frequency vector $\underline{\omega}$ are both N-dimensional row vectors, with a spacing of t_δ and ω_δ respectively.
The case of a Gaussian embedding for the algorithm with a PDF for the impulse response is implemented (§3.2.3) because of the simple form of the log likelihood function (3.5) and the availability of the *Fisher information matrix* like described in §3.2.4.
Here a few notes will be given on the implementation of the different terms of the algorithm of paragraph 3.2.

4.2.1 $G_0(e^{-j\omega}, \Theta_0) = \Lambda(e^{-j\omega})\Theta_0$

The model has to be parametrized linearly in the parameters. Θ_0 only contains the parameters of the numerator. The estimated parameters of the denominator are included in Λ and so these parameters are left out of the computation of the model error. In the implementation the standard ARX identification procedures are used. This is a more practical solution than the use of Laguerre polynomials such as in [Goodwin 92], but the resulting denominator is considered to be good. In fact a lot of faith is put into the quality of the denominator. This a disadvantage of this implementation, but it is in general a problem with this algorithm.
Λ is a *row* vector of length p and is computed for each frequency sample ω_k.

4.2.2 $G_\Delta(e^{-j\omega}) = \Pi(e^{-j\omega})\eta$

Since the frequency vector is included in a DFT operation, the frequency vector has to be in the interval from 0 to 2π otherwise aliasing will occur.
The implementation of Π is straightforward. Mathematically Π is a row of length L, where L is the order of the FIR model by which the undermodelling error can be approximated. In the implementation L=N like in §3.2.3. In Matlab for each frequency sample ω_k this row has to be computed so the implemented Π is a NxN matrix.
Also the implementation of the regression vector $\Phi(\underline{k})$ with shifted versions of the input is straightforward. The regression vector with respect to the undermodelling $\Psi(\underline{k})$ is constructed as a mirrored Hankel matrix.

4.2.3 Estimation of the covariance parameters

For the maximization of the log likelihood function (3.5) the Matlab 'Optimization toolbox' is used. A new data vector W with a non-singular distribution has to be constructed and this can be done according to (3.4) $W = R^T Y$, with R any N-p independent linear combination of the columns of P, which is the orthoplement of regression vector Φ. The construction of R and W is straightforward with the expressions of §3.2.3.

The constant in the log likelihood function has no influence on the values of the estimated parameters and is taken equal to 0.
A good choice of the initial values of the covariance parameters $\xi_0 = [\alpha_0, \lambda_0, \sigma^2_{v,0}]$ which have to be estimated is essential for a good and fast result. This is mainly a trial and error procedure. The optimization procedure is time-consuming and it can not be guaranteed that the true global maximum will be found. The algorithm can converge to a local maximum. Furthermore, it is in general not possible to estimate the covariance parameter of the noise $\gamma = \sigma^2_v$. When the parameters $\hat{\xi} = [\hat{\alpha}, \hat{\lambda}, \hat{\sigma}^2_v]$ are estimated then the estimation of the model error $\mathcal{E}\{|\hat{G}_{\Delta,N}(\omega)|^2\}$ can be computed according to theorem 3.1b and using the expressions for the covariances of §3.2.3. In the m-file the undermodelling error is denoted with Gd1 and the noise error with Gd2.

4.3 Implementation of the algorithm with a prior PDF for the frequency function

4.3.1 The routine *goodwin1.m*

If in the main programme file *goodwin.m* the error-algorithm with the PDF for the frequency function according to [Goodwin 89] is chosen, then the m-file *goodwin1.m* is called. In this m-file the notation of Goodwin is used as much as possible.
First a second part of the initialisation takes place. The process G_Δ is assumed to be a wide-sense stationary process and then the PDF of the frequency function G_Δ is specified in the form of a covariance function which depends only on the difference of the frequency samples. A stationary or non-stationary description of the covariance can be chosen, as described in chapter 3 and then the corresponding parameters have to be specified. The covariance function is plotted as a function of $\underline{\Delta\omega} = \underline{\omega}_1 - \underline{\omega}_2 = [-\omega_{2,max},\ldots,\omega_{1,max}]$. $\underline{\omega}_1$ and $\underline{\omega}_2$ are row vectors of length N and $\underline{\omega}_1$ and $\underline{\omega}_2$ start at a frequency of $\omega_\delta = \omega_{max}/N$ rad/s. The frequency vectors do not start at 0 because then 1/0 has to be computed in for example $P(j\omega)/D(j\omega)$ and this would result in a warning and NaN's (Not a Number) in Matlab.
In case of a nominal ARX model the parameters of the model are necessary and a filtered input is used in the error algorithm: $U_f(s) = \hat{A}_N(s)/F(s) \bullet U(s)$. In case of a FIR model the undermodelling error can be computed without the need of an estimated model.

$\left.\frac{\partial G^T}{\partial \Theta}\right|_{\Theta=\hat{\Theta}_N}$ is replaced by $P(j\omega)$ and no additional filter is used so $U_f(s) = U(s)$.

Before the expectation of the model error $\hat{G}_{\Delta,N}$ is computed, the polynomial D(s) is shown to the user. D(s) will be used to make the computation of the regression vector more numerically stable, as described in §3.3.2.

In the m-files the following notation is used for the different transfer functions:
- the difference between the estimated model and the nominal model $\hat{G}_N - G_0$ = Gd3;
- the transfer function of the a priori modelling error $G_{\Delta,0}$ = Gd0;
- the a posteriori modelling error $\hat{G}_{\Delta,N}$ = GdN.

According to theorem 3.2, the difference between the true system transfer function and the optimal/final estimated one, satisfies:

$$\mathcal{E}\{|\hat{G}_{\Delta,N}|^2\} = \mathcal{E}\{|GdN|^2\} = \mathrm{cov}\{G_{\Delta,0}(\omega,\omega)\} + \mathcal{E}\{|\hat{G}_N - G_0|^2\} - 2\mathrm{Re}\mathcal{E}\{(\hat{G}_N - G_0)G_{\Delta,0}^*\}$$
$$= \mathrm{cov}(Gd0(0)) + \mathcal{E}\{|Gd3|^2\} - 2\mathrm{Re}\mathcal{E}\{Gd3Gd0^*\}.$$

So the three terms in this equation have to be computed and this is done in the m-file *goodwn1b.m.*

4.3.2 The prior estimate of the undermodelling

$cov\{GdO(\omega,\omega)\} = \overline{cov}\{GdO(\omega-\omega)\} = \frac{\sigma_0^2}{\beta}$, the last result only holds when the stationary covariance (3.6) for the undermodelling is used. This is a dc-component.

4.3.3 $\mathcal{E}\{|\hat{G}_N - G_0|^2\} = \mathcal{E}\{|Gd3|^2\}$

The regression vector, the filter for numerical implementation included, becomes:

$$\Phi(\underline{k}) = \frac{\left[P^m(\rho_k)^T \mid P^{n-1}(\rho_k)^T\right]^T}{d(\rho_k)} \begin{bmatrix} \underline{u_f}(\underline{k}) \\ \hline \underline{y}(\underline{k}) \end{bmatrix} = \left[\frac{\rho_k^m \underline{u_f}(\underline{k})}{d(\rho_k)} \cdots \frac{\underline{u_f}(\underline{k})}{d(\rho_k)} \frac{\rho_k^{n-1}\underline{y}(\underline{k})}{d(\rho_k)} \cdots \frac{\underline{y}(\underline{k})}{d(\rho_k)} \right]^T .$$

m is the order of the B polynomial (numerator) and n the order of the A polynomial (denominator) of the model. Each element of Φ itself is a row vector of length N. The combination of the differentiation and filter D is implemented as a filtering operation (i.e. the derivatives are computed using a first order forward method). In the implementation the zero mean property of the input and output signals and so of their derivatives and primitives is used. If the mean of a signal appearing in the regression vector is not equal to zero, then this offset is corrected. The 0-th, first and second order operations go well in practice. The results of higher order operations deteriorate!

Define

$$p_1(\underline{k}) := \Phi(\underline{k})\Phi^T(\underline{k}) = \begin{bmatrix} \frac{(\rho_k^m u_f(\underline{k}))^2}{d(\rho_k)^2} & \cdots & \frac{\rho_k^m u_f(\underline{k}) u_f(\underline{k})}{d(\rho_k)^2} & \frac{\rho_k^{m+n-1} u_f(\underline{k}) y(\underline{k})}{d(\rho_k)^2} & \cdots & \frac{\rho_k^m u_f(\underline{k}) y(\underline{k})}{d(\rho_k)^2} \\ \vdots & & \vdots & \vdots & & \vdots \\ \frac{\rho_k^m y(\underline{k}) u_f(\underline{k})}{d(\rho_k)^2} & \cdots & \frac{y(\underline{k}) u_f(\underline{k})}{d(\rho_k)^2} & \frac{\rho_k^n y(\underline{k}) y(\underline{k})}{d(\rho_k)^2} & \cdots & \frac{y(\underline{k})^2}{d(\rho_k)^2} \end{bmatrix}.$$

Each entry is a row of length N so $p_1(\underline{k})$ is a (m+1+n)x(m+1+n)N matrix. P is the inverse of the matrix with integrals (summations) of each entry of the p_1 matrix. P is a (m+1+n)x(m+1+n) matrix:

$$P = \left(\frac{1}{T} \sum_{k=1}^{N} p_1(\underline{k}) \delta k \right)^{-1} = \begin{bmatrix} P_{1,1} & \cdots & P_{1,m+1+n} \\ \vdots & & \vdots \\ P_{m+1+n,1} & \cdots & P_{m+1,m+1+n} \end{bmatrix}.$$

This inverse operation is a risk when it's operand has a bad condition (it is almost singular). If this occurs, the user receives a warning of this and this happens quit often in practice. (Pay attention to the fact that also 1/T is included in the inverse-operation!)

Then

$$M_1(\underline{\omega}_1, \underline{k}_2) = \frac{1}{N} \sum_{l=1}^{N} \overline{cov}\{Gd0(\underline{\omega}_2^l - \underline{\omega}_1)\} U(\underline{\omega}_2^l) e^{j\underline{\omega}_2^l \underline{k}_2} \qquad with\ \underline{\omega}_2^l = \frac{2\pi l}{N} \quad (l = 0,...,N-1)$$

the Inverse Discrete Fourier Transform (IDFT) of $\underline{\overline{cov}}\{Gd0(\underline{\omega}_2 - \underline{\omega}_1)\}U(\underline{\omega}_2)$, where $U(\underline{\omega}_2)$ is the DFT of input $u(\underline{k})$ for the discrete frequencies $\underline{\omega}_2^1$. So $\underline{\omega}_2 = (0, 2\pi]$.
Remark: the *superscripts* denote the indices of the elements of a vector or matrix.

$\underline{\overline{cov}}\{Gd0(\underline{\omega}_2 - \underline{\omega}_1)\}$ is a complex NxN frequency matrix of which the rows are the ω_1 direction and the columns in the ω_2 direction. $\underline{\omega}_1$ will also be defined through a DFT operation (see below) so $\underline{\omega}_1 = (0, 2\pi]$. The input frequency samples $\underline{\Delta\omega}$ can be put into a matrix

$$\underline{\Delta\omega} = \begin{bmatrix} \underline{\omega}_2^1 - \underline{\omega}_1^1 & \cdots & \underline{\omega}_2^N - \underline{\omega}_1^1 \\ \vdots & & \vdots \\ \underline{\omega}_2^1 - \underline{\omega}_1^N & \cdots & \underline{\omega}_2^N - \underline{\omega}_1^N \end{bmatrix} = \begin{bmatrix} 0 & \cdots & 2\pi - \omega_\delta \\ \vdots & & \vdots \\ -2\pi + \omega_\delta & \cdots & 0 \end{bmatrix}$$ and for these frequencies the

covariance function is calculated.

Pay attention to the use of the DFT or FFT algorithms (and their inverses), see appendix B. In computing $M_1(\underline{\omega}_1, \underline{k}_2)$ the function $\underline{\overline{cov}}\{Gd0(\underline{\omega}_2 - \underline{\omega}_1)\}$ has to be limited to an interval from 0 to 2π over which the inverse Fourier transformation is computed. The problem is that the covariance functions as defined in chapter 3, are not bounded. The bounding of the covariance has been taken care of in the implementation of the covariance function in the m-file *cova.m.*
- By putting a window of width 2π over the covariance function and eliminating the parts of the covariance which lie outside the window the covariance function can be bounded. Unfortunately hereby the high frequency information is lost so the resulting IFFT is only an approximation and this will show up in the final results.
- In the FFT algorithm of Matlab, the Fourier transform is computed for frequencies between

0 and 2π rad/s instead of between $-\pi$ and $+\pi$, which is the usual interval for a DFT operation. This makes no difference if the function is periodical in the frequency domain, but the covariance function is not, so in the implementation the part of the spectrum from $-\pi$ to 0 is moved to the frequency interval $[\pi, 2\pi]$.
For M_1 the IDFT is computed for the different samples of $\underline{\omega}_1$ (the rows of M_1 are the frequency points and the columns are in time domain, M_1 is a NxN matrix). A frequency sample of $\underline{\omega}_1(\neq 0)$ performs a shift of $\overline{\text{cov}}\{\text{Gd0}\}$ over the ω_2-axis. So this inverse Fourier operation for the different samples of $\underline{\omega}_1$ could be regarded as moving a window of width 2π over the covariance as function of $\underline{\omega}_2$, starting at $0 \le \underline{\Delta\omega} \le 2\pi$ and finishing at $2\pi \le \underline{\Delta\omega} \le 4\pi$.

Remark: Theoretically / mathematically the time vectors $\underline{k}_2$ and $\underline{k}_1$ ($\underline{k}_1$ will be defined below) are different from each other and different from $\underline{k}$. $\underline{k}_2$ results from an IDFT operation and $\underline{k}_1$ from a DFT operation. However in the implementation the different time vectors span the same interval ($\underline{k}_2 = \underline{k} = \underline{k}_1$) .

Define

$$Q_1(\underline{k}_1,\underline{k}_2) = \frac{1}{N}\sum_{k_1=1}^{N} M_1(\underline{\omega}_1,\underline{k}_2)U^*(\underline{\omega}_1)e^{-j\underline{\omega}_1\underline{k}_1} \qquad \text{the DFT of } 1/N\bullet M(\underline{\omega}_1,\underline{k}_2)U^*(\underline{\omega}_1), \text{ a NxN}$$

2-dimensional time space matrix.

$$p_2(\underline{k}_2,\underline{k}_1) := \Phi(\underline{k}_2)\Phi^T(\underline{k}_1) =$$

$$\frac{1}{d(\rho_k)^2}\begin{bmatrix} \rho_k^{2m}u_f(\underline{k}_2)u_f(\underline{k}_1) & \cdots & \rho_k^{m}u_f(\underline{k}_2)u_f(\underline{k}_1) & \rho_k^{m+n-1}u_f(\underline{k}_2)y(\underline{k}_1) & \cdots & \rho_k^{m}u_f(\underline{k}_2)y(\underline{k}_1) \\ \vdots & & \vdots & \vdots & & \vdots \\ \rho_k^{m}y(\underline{k}_2)u_f(\underline{k}_1) & \cdots & y(\underline{k}_2)u_f(\underline{k}_1) & \rho_k^{n}y(\underline{k}_2)y(\underline{k}_1) & \cdots & y(\underline{k}_2)y(\underline{k}_1) \end{bmatrix}$$

and this is a (m+1+M)Nx(m+1+M)N matrix. Then finally the second term of the theorem results:

$$\mathcal{E}\{|Gd3|^2\} = \left(\frac{1}{D(j\underline{\omega})}\frac{\partial G(j\underline{\omega})}{\partial\Theta}\Big|_{\Theta=\hat{\Theta}_N}\right)^T \cdot$$

$$P\ \frac{1}{T^2}\sum_{j=1}^{N}\sum_{l=1}^{N} p_2(\underline{k}_2(l),\underline{k}_1(j))Q_1(\underline{k}_1(j),\underline{k}_2(l))\,\delta\underline{k}_1\delta\underline{k}_2\ P\left(\frac{\partial G(j\underline{\omega})}{\partial\Theta}\Big|_{\Theta=\hat{\Theta}_N}\frac{1}{D(j\underline{\omega})}\right)^*.$$

$p_2Q_1(\underline{k}_1,\underline{k}_2)$ is a (m+1+n)N×(m+1+n)N two-dimensional time matrix. After one integration a (m+1+n)×(m+1+n)N matrix results and one time dimension has disappeared. After the second integration a (m+1+n)×(m+1+n) matrix results. This double summation is the most time consuming part of the algorithm GOODWIN1.
For the implementation of the term $\partial G/\partial\Theta$ expression (3.7) can be used and only the filter

D(s) has to be added. In case of an ARX model $\partial G/\partial\Theta$ consists of m+1+n transfer functions, which are calculated for all frequency samples of $\underline{\omega}$. This is done with the Matlab function 'freqs'.
$\mathcal{E}\{|Gd3|^2\}$ is a frequency domain vector of length N.

4.3.4 $2Re\mathcal{E}\{(\hat{G}_N - G_0)G_{\Delta,0}^*\} = 2Re\mathcal{E}\{Gd3Gd0^*\}$

$$M_2(\underline{\omega},\underline{k_2}) := \frac{1}{N}\sum_{l=1}^{N} \overline{cov}\{Gd0(\underline{\omega_2^l}-\underline{\omega})\}\, U(\underline{\omega_2^l})\, e^{j\underline{\omega_2^l}\underline{k_2}} \qquad \text{the IDFT of } \overline{\text{cov}}\{\text{Gd0}(\underline{\omega_2}-\underline{\omega})\}$$

($\underline{\omega}$ is used instead of $\underline{\omega_1}$ like in M_1) and the Fourier transform $U(\underline{\omega_2})$ of the input $u(\underline{k})$. $M_2(\underline{\omega},\underline{k_2})$ is a NxN matrix in frequency and time domain (the rows are the frequency points and the columns are in time domain).
Define

$$q_2(\underline{\omega},\underline{k_2}) := \Phi(\underline{k_2})M_2(\underline{\omega},\underline{k_2}) = \begin{bmatrix} \Phi_2^1 \\ \vdots \\ \Phi_2^{m+1+n} \end{bmatrix} M_2 = \begin{bmatrix} \Phi_2^1 M_2 \\ \vdots \\ \Phi_2^{m+1+n} M_2 \end{bmatrix} = \begin{bmatrix} q_2^1 \\ \vdots \\ q_2^{m+1+n} \end{bmatrix}.$$

Each of the m+1+n time functions / rows of Φ_2 have to be multiplied by the NxN frequency-time matrix M_2 so this results in a (m+1+n)NxN matrix. From each of the m+1+n blocks $q_2^i(\underline{\omega},\underline{k_2})$ with size NxN, the time integral (i.e. the integral with respect to the different rows of each block) has to be computed:

$$Q_2(\underline{\omega}) := \frac{1}{T}\sum_{j=1}^{N} q_2(\underline{\omega},\underline{k_2}(j))\delta\underline{k_2} = \begin{bmatrix} Q_2^1 \\ \vdots \\ Q_2^{m+1} \end{bmatrix}.$$

By putting the resulting frequency samples for the different integrated blocks of q_2 as rows of Q_2, this becomes a (m+1+n)xN frequency matrix.

Then the last term of the theorem is:

$$\mathcal{E}\{Gd3Gd0^*\} = \left(\frac{1}{D(j\underline{\omega})}\frac{\partial G}{\partial\Theta}\Big|_{\Theta=\hat{\Theta}_N}\right)^T PQ_2(j\underline{\omega}).$$

P is a (m+1+n)×(m+1+n) matrix so for each frequency the product of P and a column of $Q_2(j\underline{\omega})$ is taken to form $PQ_2(j\underline{\omega})$, a (m+1+n)xN matrix. Finally $\mathcal{E}\{Gd3Gd0^*(\underline{\omega})\}$ is a row vector of length N.

Now -2 times the real part of $\mathcal{E}\{Gd3Gd0^*\}$ has to be taken and then the expectation of the model error $\mathcal{E}\{|\hat{G}_{\Delta,N}|^2\} = \mathcal{E}\{|GdN|^2\}$ can be computed and plotted as a function of ω.

In case of a nominal FIR model $\partial G/\partial\Theta$ can be replaced by $P(j\omega)$, a vector of decreasing powers of s, starting with s^m and ending at $1 = s^0$.
In case of 0-th (a constant) and 1-st order FIR models, the expectation of the undermodelling error can be analytically determined, as will be shown in §5.1. If the error-algorithm is used for these models then also these analytical results will be computed and plotted as a reference. These analytical results formed the basis of the verification of the numerical algorithms during implementation.

4.3.5 A note on the use of filters with ARX models

The estimation of ARX models is based on a *equation error criterion*, which is visualised in figure 4.1.
$\hat{A}_N y - \hat{B}_N u = \varepsilon_0$ with ZMW output noise e, so the goal of the Least Square procedure is to minimize the expectation of the energy of the equation error ε_0, which comes down on making it ZMWN.

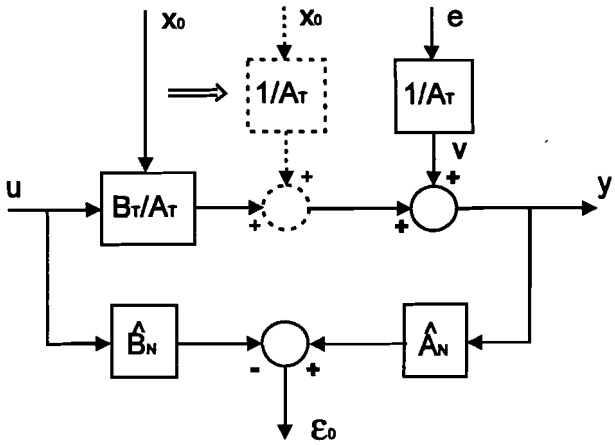

figure 4.1: The equation error algorithm

When a filter 1/D is added to the regression vector, then the initial conditions x_0 of the filter also are an input of the true system, as can be seen in figure 4.1 and these initial conditions will appear in the output y. The initial conditions can be regarded as output noise, which is visualised with the dotted box and arrows in figure 4.1. This filtered version of x_0 will in general *not* be ZMWN. This signal is included in the output y and so it will also be included in the Least Squares algorithm. When this algorithm makes the equation error ZMWN, then never the true system B_T/A_T will be identified. This is a major problem when we try to make the numerical implementation of the regression vector better by the use of a filter. However, if the filter D is stable then after some time the initial conditions will have decayed. So if we use the last N samples of a data-set which is (much) longer than N, then the effects of the initial filter conditions will be negligible and this results in the best possible ARX model. This is also the way in which it is done in the m-file *regres1.m*.

5. Examples with the algorithms for model errors in a stochastic embedding

In this chapter the implementation of the algorithms of chapter 3 is tested in a few examples.. First the examples of the articles [Goodwin 89] and [Goodwin 92] will be used. After this, a practical process will be regarded.

5.1 Examples of the error algorithm with FIR models

For 0-th and 1-st order FIR models, the analytical solution for the expectation of the model error can be computed as presented in [Goodwin 89]. These results can be used as a reference and this has been done during the implementation.

5.1.1 0-th order FIR model (m=0)

First, for a 0-th order nominal model the analytical solution is derived starting from theorem 3.2.

The system consists of a simple gain together with the unmodelled dynamics, i.e. $G_T(j\omega) = b0 + G_\Delta(j\omega)$. Because the estimated model is a FIR, the undermodelling error is not a function of the estimated parameters so $\frac{\hat{A}_N}{A_0} = 1$ and no filter F(s) is used. For the filter polynomial in the regression vector D(s) = 1 is used.

With a sinusoidal input $u = \cos(\omega_0 t)$ then $\Phi(t) = \cos(\omega_0 t)$. It is convenient that $\frac{1}{D(j\omega)}\frac{\partial G(j\omega)}{\partial \Theta} = P(j\omega) = 1$. So then

$$\hat{G}_N(j\omega) - G_0(j\omega) = \frac{\hat{A}_N}{A_0}\frac{\partial G(j\omega)}{\partial \Theta}(\hat{\Theta}_N - \Theta_0) = \hat{\Theta}_N - \Theta_0 = P\,\frac{1}{T}\int_0^N \Phi(t)\eta(t)dt = Re\{G_{\Delta,0}(j\omega_0)\},$$

because the impulse response of the undermodelling $\eta(t)$ is related to u(t) by the frequency function of the undermodelling $G_\Delta(j\omega)$.

So $$|\hat{G}_{\Delta,N}(\omega)|^2 = |G_{\Delta,0}(\omega)|^2 + |Re\{G_{\Delta,0}(\omega_0)\}|^2 - 2Re\{G_{\Delta,0}(\omega_0)G_{\Delta,0}(\omega)^*\}.$$

Taking the expectation of the terms in the above equation leads to:

$$\mathcal{E}\{|\hat{G}_{\Delta,N}(\omega)|^2\} = \overline{cov}G_{\Delta,0}(\omega-\omega) + Re\{\overline{cov}G_{\Delta,0}(\omega_0)\}^2 - 2Re\{\overline{cov}G_{\Delta,0}(\omega_0)\overline{cov}G_{\Delta,0}(\omega)^*\} =$$

$$= \overline{cov}\{G_{\Delta,0}(0)\} + \frac{1}{2}Re\left[\overline{cov}\{G_{\Delta,0}(0)\} + \overline{cov}\{G_{\Delta,0}(2\omega_0)\}\right] - Re[\overline{cov}\{G_{\Delta,0}(\omega+\omega_0)\} +$$

$$\overline{cov}\{G_{\Delta,0}(\omega-\omega_0)\}] = \frac{\sigma_0^2}{\beta} + 0.5\frac{\sigma_0^2}{\beta} + 0.5\frac{\sigma_0^2\beta}{4\omega_0^2+\beta^2} - \frac{\beta\sigma_0^2}{(\omega+\omega_0)^2+\beta^2} - \frac{\beta\sigma_0^2}{(\omega-\omega_0)^2+\beta^2}.$$

The stationary description (3.6) of the covariance of the undermodelling has been substituted:

$$Re\left\{\frac{\sigma_0^2}{j\omega+\beta}\right\} = Re\left\{\frac{\sigma_0^2(j\omega-\beta)}{-\omega^2-\beta^2}\right\} = Re\left\{\frac{\sigma_0^2(\beta-j\omega)}{\omega^2+\beta^2}\right\} = \frac{\sigma_0^2\beta}{\omega^2+\beta^2}.$$

The following lemma's have been used:

$$\overline{cov}^*\{G_\Delta(\omega_2-\omega_1)\} = \overline{cov}\{G_\Delta(\omega_1-\omega_2)\}$$

$$\mathcal{E}\{G_\Delta(j\omega_1)G_\Delta(j\omega_2)\} = \overline{cov}\{G_\Delta(\omega_1+\omega_2)\}$$

$$\mathcal{E}\{Re[G_\Delta(j\omega_1)]Re[G_\Delta(j\omega_2)]\} = \frac{1}{2}Re\left[\overline{cov}\{G_\Delta(\omega_1+\omega_2)\} + \overline{cov}\{G_\Delta(\omega_1-\omega_2)\}\right].$$

5.1.2 First order FIR model (m=1)

According to [Goodwin 89] the analytical solution for a first order FIR model is:
$\mathcal{E}\{|\hat{G}_{\Delta,N}|^2\} = \mathcal{E}\{|G_{\Delta,0}|^2\} + \mathcal{E}\{|\hat{G}_N - G_0|^2\} - 2Re\mathcal{E}\{(\hat{G}_N - G_0)G_{\Delta,0}^*\}$, with

$\mathcal{E}\{|G_{\Delta,0}(\omega)|^2\} = \frac{\sigma_0^2}{\beta}$ and

$$\mathcal{E}\{|\hat{G}_{\Delta,N}(\omega)-G_{\Delta,0}(\omega)|^2\} = \mathcal{E}\{|Gd3(\omega)|^2\} = 0.5\frac{\sigma_0^2}{\beta}\left[1+\left(\frac{\omega_0}{\omega}\right)^2\right] + 0.5\frac{\sigma_0^2\beta}{4\omega_0^2+\beta^2}\left[1-\left(\frac{\omega_0}{\omega}\right)^2\right]$$

$$-2Re\left\{\mathcal{E}\{|\hat{G}_N(\omega)-G_0(\omega)|^2 G^*_{\Delta,0}\}\right\} = -\left(1+\frac{\omega_0}{\omega}\right)\frac{\beta\sigma_0^2}{(\omega-\omega_0)^2+\beta^2} - \left(1-\frac{\omega_0}{\omega}\right)\frac{\beta\sigma_0^2}{(\omega+\omega_0)^2+\beta^2}.$$

The results of the implemented algorithm will be compared with the analytical solutions of the different terms as much as possible. The numerical versions will be denoted with a hat ^ or with the notation which is used in the m-files and which has been introduced in paragraph 4.3.

example 5.1 The situation like in Example 2 of [Goodwin 89] is considered:
- A first order FIR model is used and because the actual (estimated) model is not relevant, each system can be used for the true system.
- The input is a sinusoidal signal $u = \cos(\omega_0 \underline{k})$, with $f_0 = 4$ Hz;
- No output noise so $\sigma_v^2 = 0$;
- A stationary covariance function with $\sigma_0^2 = \beta = 10$.

The parameters for the numerical implementation are:
- $N = 64$;
- $f_s = 50$ Hz;
- The final time $T = N \bullet t_\delta = 1.3$ s so time vector $\underline{k} = [0, 1.3]$ s;
- The frequency vector $\underline{\omega} = [0, 10 \bullet \omega_0] = [0, 251]$ rad/s.

I) $$cov\{G_{\Delta,0}(\omega,\omega)\} = \overline{cov}\{G_{\Delta,0}(\omega-\omega)\} = \frac{\sigma_0^2}{\beta}.$$

In the implementation $covGd1(\omega) = 1$, so this is right.

II) $u(\underline{k}) = \cos(\omega_0 \underline{k})$. Filter polynomial $D(s) = s$ is used, which is a pure differentiator and not strictly stable. The regression vector Φ only contains versions of the input signal and the first term

is the input itself and the second term is the primitive of the input: $$\Phi(\underline{k}) = \begin{bmatrix} \cos(\omega_0 \underline{k}) \\ \frac{1}{\omega_0}\sin(\omega_0 \underline{k}) \end{bmatrix}.$$

$$p_1(\underline{k}) = \Phi(\underline{k})\Phi^T(\underline{k}) = \begin{bmatrix} \cos(\omega_0\underline{k}) \\ \frac{1}{\omega_0}\sin(\omega_0\underline{k}) \end{bmatrix} \begin{bmatrix} \cos(\omega_0\underline{k}) & \frac{1}{\omega_0}\sin(\omega_0\underline{k}) \end{bmatrix} =$$

$$= \begin{bmatrix} \cos(\omega_0\underline{k})^2 & \frac{1}{\omega_0}\sin(\omega_0\underline{k})\cos(\omega_0\underline{k}) \\ \frac{1}{\omega_0}\sin(\omega_0\underline{k})\cos(\omega_0\underline{k}) & \frac{1}{\omega_0^2}\sin(\omega_0\underline{k})^2 \end{bmatrix}$$

The matrix p_1 can be verified, using the file *p1_ana.m.* Although for the primitivation and differentiation operations only first order methods are used the 4 parts of the numerical version of p_1 are in good agreement with the theoretical results as presented above.

$$P = \lim_{T\to\infty}\left(\frac{1}{T}\int_0^T p_1(\underline{k})d\underline{k}\right)^{-1} = \begin{bmatrix} 2 & 0 \\ 0 & 2\omega_0^2 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 1257 \end{bmatrix}.$$

The numerical version is not bad:

$$\hat{P} = \begin{bmatrix} 2.3 & 0 \\ 0 & 1417.3 \end{bmatrix}.$$

$u_2 = \cos(\omega_0\underline{k}_2) = u$ since $\underline{k} = \underline{k}_2$ and it's Fourier transform is $U_2(\underline{\omega}_2) = \pi(\delta(\underline{\omega}_2-\omega_0) + \delta(\underline{\omega}_2+\omega_0))$ where δ denotes a Dirac pulse. $\underline{\omega}_2 = [0, 2\pi] = \underline{\omega}_1$

With Maple it follows that the inverse Fourier transform of $\overline{cov}\{G_{\Delta,0}(\underline{\omega}_2-\underline{\omega}_1)\}$ and U_2 is

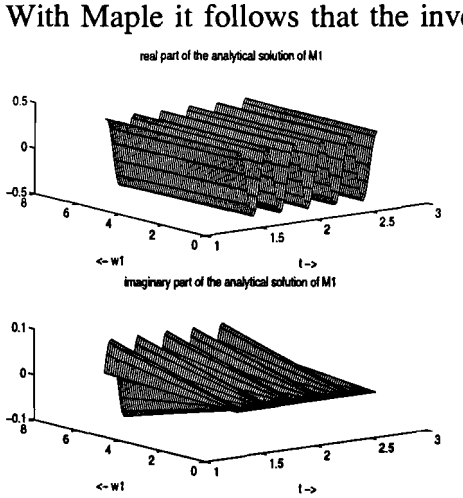

figure 5.1: Real and imaginary part of $M_1(\underline{\omega}_1,\underline{k}_2)$

$$M_1(\underline{\omega}_1,\underline{k}_2) = \frac{1}{2}\sigma_0^2\left(\frac{e^{j\omega_0\underline{k}_2}}{j(\omega_0-\underline{\omega}_1)+\beta} + \frac{e^{-j\omega_0\underline{k}_2}}{-j(\omega_0+\underline{\omega}_1)+\beta}\right).$$

This function looks like the one plotted in figure 5.1 and can be found in *m1_ana.m.*

Basically this is a cosine function in the time domain (formed by the exponential terms in the numerators) with an envelop in the frequency space related to the covariance function. The rows of M_1 are the different frequency points and the columns are samples in the time domain.

For the term Q_1 the Fourier transform of the product of

$M_1(\underline{\omega}_1,\underline{k}_2)$ and $U_1^*(j\underline{\omega}_1)$ (the Fourier transform of $u_1 = \cos(\omega_0\underline{k}_1)$) has to be computed. Since both M_1 and U_1^* are related to a cosine function it is likely that the 2-dimensional function $Q_1(\underline{k}_1,\underline{k}_2)$ will consist of a kind of cosine function in both time dimensions.

$$Q_1(\underline{k}_1,\underline{k}_2) = \frac{1}{2\pi}\int_{-\infty}^{\infty} M_1(\underline{\omega}_1,\underline{k}_2)\pi\left(\delta(\underline{\omega}_1-\omega_0)+\delta(\underline{\omega}_1+\omega_0)\right)e^{-j\underline{\omega}_1\underline{k}_1}d\underline{\omega}_1 =$$

$$\frac{1}{4}\sigma_0^2\left(\frac{e^{j\omega_0\underline{k}_2}}{j(\omega_0-\omega_0)+\beta} + \frac{e^{-j\omega_0\underline{k}_2}}{-j(\omega_0+\omega_0)+\beta}\right)e^{-j\omega_0\underline{k}_1} = \frac{1}{4}\sigma_0^2\left(\frac{e^{j\omega_0\underline{k}_2}}{\beta} + \frac{e^{-j\omega_0\underline{k}_2}}{\beta-2j\omega_0}\right)e^{-j\omega_0\underline{k}_1}.$$

The form of the numerical result of Matlab is in agreement with this assumption. However the analytical amplitude of Q_1 is 0.26 and the numerical amplitude is 1.02!
$p_2(\underline{k}_1,\underline{k}_2) = \Phi_2(\underline{k}_2)\Phi_1^T(\underline{k}_1)$ and consists of 4 NxN blocks in the 2 time dimensions k_1 and k_2. The analytical and numerical version of p_2 can be viewed with *p2Q1_ana.m*. At first sight the structure, period and amplitude of the numerical and analytical version of p_2 are the same. Then each block/function of p_2 has to be multiplied by Q_1 which is also a function of the 2 time dimensions:

$$p_2Q_1(\underline{k}_1,\underline{k}_2) = \begin{bmatrix}[p_2Q_1]^{11} & [p_2Q_1]^{12}\\ [p_2Q_1]^{21} & [p_2Q_1]^{22}\end{bmatrix} = \begin{bmatrix}p_2^{11}Q_1 & p_2^{12}Q_1\\ p_2^{21}Q_1 & p_2^{22}Q_1\end{bmatrix}$$

and this remains a 2Nx2N time matrix. The m-file *p2Q1_ana.m* shows that each function/block $p_2^{ij}Q_1^{ij}$ is the product of sinusoidal functions.
Of this matrix a double integral with respect to $\underline{k}_1$ and $\underline{k}_2$ has to be computed. First the integral to $\underline{k}_1$ is taken and this results in a 2Nx2 matrix in the $\underline{k}_2$ space. Finally the integral with respect to $\underline{k}_2$ of each integral-term/block is computed and a 2x2 matrix results. This

analytical matrix is $\begin{bmatrix}.11354+.02062j & .00102-.00438j\\ .00102+.00406j & .00016-.00003j\end{bmatrix}$

and the numerical version $INTINTp2Q1 = \begin{bmatrix}0.1943+0.0787j & -0.0009-0.0015j\\ 0.0001+0.0002j & 0.0000-0.0000j\end{bmatrix}$.

The difference between the analytical and numerical version is large.

The second term of the theorem follows:

$$\mathcal{E}\{|Gd3(\underline{\omega})|^2\} = \left(\frac{P(j\underline{\omega})}{D(j\underline{\omega})}\right)^T P[INTINTp2Q1]P\left(\frac{P(j\underline{\omega})}{D(j\underline{\omega})}\right)^* =$$

$$= \begin{bmatrix} 1 & \frac{1}{j\underline{\omega}} \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 2\omega_0^2 \end{bmatrix} \begin{bmatrix} INTINTp_2Q_1^{11} & INTINTp_2Q_1^{12} \\ INTINTp_2Q_1^{21} & INTINTp_2Q_1^{22} \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 2\omega_0^2 \end{bmatrix} \begin{bmatrix} 1 \\ \frac{1}{j\underline{\omega}} \end{bmatrix}.$$

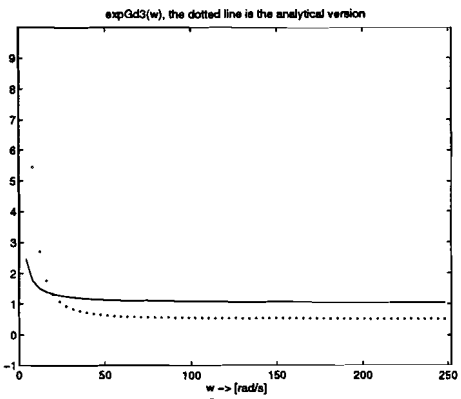

figure 5.2: $\mathcal{E}|Gd3|^2$

$\mathcal{E}\{|Gd3(\underline{\omega})|^2\}$ is a quadratic term (dominated by the two $1/j\omega$ factors). This is shown in figure 5.2. The matrices P and INTINTp2Q1 are only scalings / coefficients and the large difference between the analytical an numerical version of these coefficients results in a vertical offset.

III) The inverse fourier transform of $covGd1(\underline{\omega}_2-\underline{\omega})$ times $U_2(\underline{\omega}_2)$ results in

$$M_2(\underline{\omega},\underline{k}_2) = \frac{1}{2}\sigma_0^2\left(\frac{e^{j\omega_0\underline{k}_2}}{j(\omega_0-\underline{\omega})+\beta} + \frac{e^{-j\omega_0\underline{k}_2}}{-j(\omega_0+\underline{\omega})+\beta}\right).$$

$Q_2(j\underline{\omega})$ (which is not related to Q_1) becomes

$$Q_2(j\underline{\omega}) = \frac{1}{T}\int_0^T \Phi_2(\underline{k}_2)M_2(\underline{\omega},\underline{k}_2)d\underline{k}_2 = \begin{bmatrix} Q_2^1 \\ Q_2^2 \end{bmatrix} = \begin{bmatrix} \frac{\frac{1}{2}\sigma_0^2(\beta - j\underline{\omega})}{\omega_0^2-\underline{\omega}^2+\beta^2-2\beta j\underline{\omega}} \\ \frac{\frac{1}{2}\sigma_0^2}{\omega_0^2-\underline{\omega}^2+\beta^2-2\beta j\underline{\omega}} \end{bmatrix}$$, this is a 2NxN

matrix.

Multiplication with P leads to:

$$\overline{P}Q_2 = \begin{bmatrix} 2 & 0 \\ 0 & 2\omega_0^2 \end{bmatrix} Q_2 = \begin{bmatrix} \dfrac{\sigma_0^2(\beta - j\underline{\omega})}{\omega_0^2 - \underline{\omega}^2 + \beta^2 - 2\beta j\underline{\omega}} \\ \dfrac{\sigma_0^2\omega_0^2}{\omega_0 - \underline{\omega}^2 + \beta^2 - 2\beta j\underline{\omega}} \end{bmatrix}$$

and

$$\left(\frac{P(j\underline{\omega})}{D(j\underline{\omega})}\right)^T PQ_2 = \begin{bmatrix} 1 & \dfrac{1}{j\underline{\omega}} \end{bmatrix} PQ_2 = \begin{bmatrix} \dfrac{\sigma_0^2(\beta - j\underline{\omega})}{\omega_0^2 - \underline{\omega}^2 + \beta^2 - 2\beta j\underline{\omega}} \\ \dfrac{-j\,\sigma_0^2\omega_0^2}{\underline{\omega}(\omega_0 - \underline{\omega}^2 + \beta^2 - 2\beta j\underline{\omega})} \end{bmatrix}.$$

Then for $\mathcal{E}\{(\hat{G}_N - G_0)G_{\Delta,0}^*\}$ -2 times the real part of this expression is taken, which in fact means that the real part of PQ_2^1 is taken and the imaginary part of PQ_2^2, multiplied by $-1/\omega$. This function is plotted in figure 5.3.

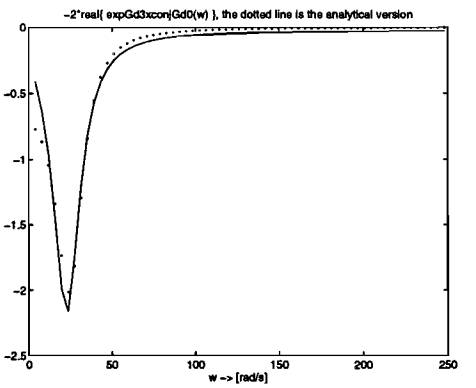

figure 5.3: Third term of the undermodelling error

The resulting expectation of the model error $\mathcal{E}\{|\hat{G}_{\Delta,0}|^2\}$ is plotted in figure 5.4. Also the results for $\omega_0 = 2\pi$ rad/s (with $f_s = 12$ Hz) and $\omega_0 = 1.25$ rad/s ($f_s = 3$ Hz) are plotted.

In figure 5.5 the undermodelling error for a 0-th order FIR model is plotted for $\omega_0 = 25$ rad/s ($f_s = 50$ Hz), $\omega_0 = 2\pi$ rad/s (with $f_s = 12$ Hz) and $\omega_0 = 1.25$ rad/s ($f_s = 3$ Hz) and the other signal, model and algorithm parameters are the same as in the rest of example 5.1.

In both cases are the numerical versions quite well in agreement with the analytical (dotted) ones. So the algorithm with a PDF for the frequency function of the undermodelling works for the simple cases. The undermodelling error has a minimum at the frequency of the signal which is used for identification as could be expected. For first and higher order models the term $\partial G/\partial\Theta$ contains $1/j\omega$ terms and so the model error goes asymptotically to infinity if the

frequency becomes 0.

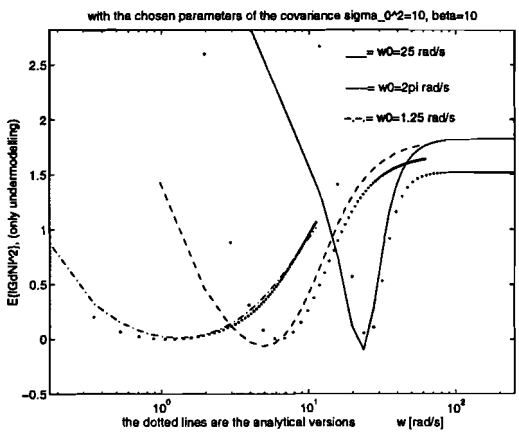

figure 5.4: The expectation of the modelling error for a first order FIR model.

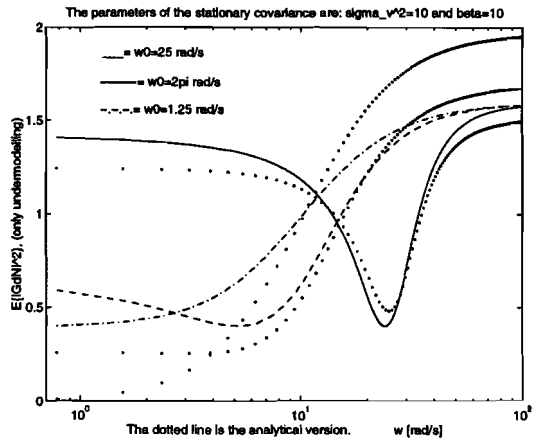

figure 5.5 The expectation of the modelling error for a constant

For a square wave input signal, the results are identical as with a sinusoidal input.
When higher order FIR models are used, the error shows more 'dynamics' as could be expected. (For example, use a second order FIR model with a sinusoidal input of 0.5Hz and a stationary description of the covariance with $\sigma_0^2 = 1$ and $\beta = 0.3$.) There is still a vertical asymptote for $\omega = 0$ rad/s, and a minimum in the expectation of the error at (near) the input signal frequency. With higher order models, the algorithm becomes time and memory consuming and when 'Windows' starts to swap to the virtual memory on the hard disc to 'increase' the memory, the time performance is really bad.

5.2 Examples of the algorithm with ARX models

example 5.2

With ARX models the undermodelling error does depend on the (parameters of) estimated model. The true system is the same as in example 3 of [Goodwin 89]:

$$G_T(s) = \frac{15}{(s+15)(s+1)} \quad .$$

- The input is a sinusoidal signal u = $2\cos(\omega_0 \underline{k})$, with f_0 = 0.8 Hz so ω_0 = 5 rad/s, which is plotted in figure 5.6;
- No output noise so $\sigma_v^2 = 0$;

The parameters for the numerical implementation are:

- N = 128;
- f_s = 37.5 Hz;
- The final time T = N•t_δ = 3.4 s so time vector $\underline{k}$ = [0, 3.4] s;
- The frequency vector $\underline{\omega}$ = [0, ω_0] = [0, 5] rad/s.

An ARX model is estimated, using the built in procedure of *goodwin.m*, with NN = [n_a,n_b,n_k] = [1,1,0], where n_a, n_b and n_k are the orders of the A- and B-polynomials and the dead time of the model respectively. In identification the filter F(s) = s+2 has been used. The resulting model is $\hat{G}_N(q) = \frac{0.0246}{q-0.9904}$ and the frequency function is plotted in figure 5.7.

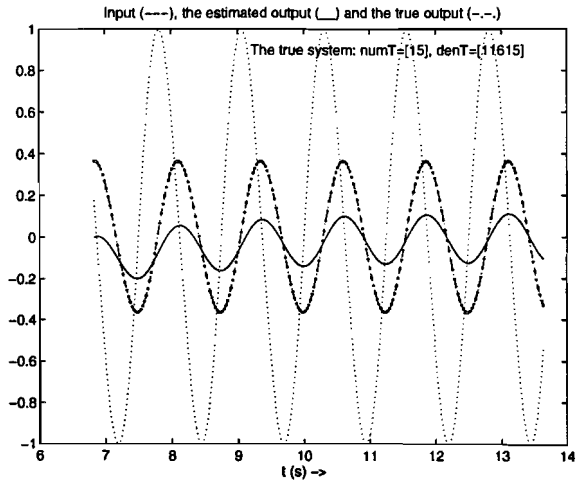

figure 5.6: The input and output signals

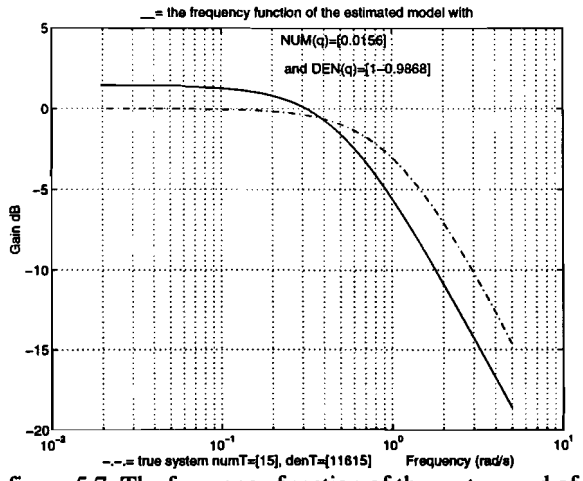

figure 5.7: The frequency function of the system and of the model

As can be seen in figure 5.7 the model matches the system exactly at a frequency of 0.4 rad/s

so there should be a minimum of 0 in the (undermodelling) error. In figure 5.7 it can also be seen that at the signal frequency of 5 rad/s the gain of model is 5dB less than the true system. This appears in the amplitude of the estimated output in figure 5.6.

For the frequency function of the undermodelling (GOODWIN1) a stationary covariance function with σ_0^2=0.2 and β=20 is used. The resulting error is plotted in figure 5.8.
For the (variance of the) impulse response of the undermodelling the following parameters have been estimated $\hat{\alpha}$=1 and $\hat{\lambda}$=0.1837 (the variance of the noise σ_v^2 is of course 0). The resulting undermodelling error can be found in figure 5.9.

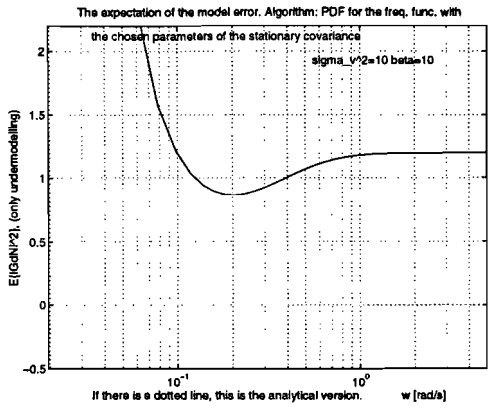

figure 5.8 The undermodelling error according to the error algorithm with a PDF for the frequency function

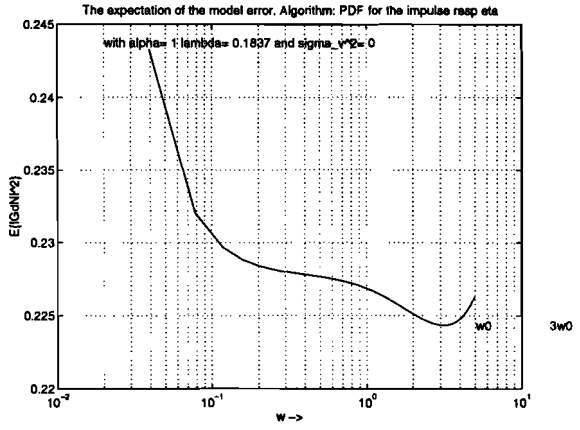

figure 5.9 The undermodelling error according to the error algorithm with a PDF for the impulse response of the undermodelling

As can be seen, both versions of the error are not in agreement with each other and are never 0. In both cases the minimum of the error does not coincide with the true minimum of the error at 0.4 rad/s. In figure 5.8 and 5.9 a vertical asymptote is visible for $\omega = 0$ as could be expected and the error is bounded (small) for higher frequencies when the gain of the true system is small (decreases). Some of the qualitative notions about the error are visible, but the quantitative results do not make sense.

example 5.3
An example as in §V of [Goodwin 92] is repeated.

The following continuous-time system was used: $G(s) = \dfrac{1}{10s^2+11s+1} = \dfrac{1}{(10s+1)(s+1)}$.

This system was simulated with sample period t_δ = 1 s. This is strange since the sample rate is *less* than the lowest allowable sample frequency according to Shannon ($\omega_s = 2\pi$ should be

larger than $2\omega_{max} = 4\pi$ rad/s) so aliasing should occur)[1].
The test input sequence $\{u_k\}$ is a 0.02Hz fundamental square wave ($\omega_0 = 0.13$ rad/s). The output of the system is corrupted with a noise sequence $\{v_k\}$ distributed as $v_k \sim N(0,0.005)$. The signals are plotted in figure 5.10.
A second order ARX model has been estimated with NN=[2, 2, 0] and filter $F(s) = 0.1s^2+s+1$:

$$\hat{G}_N(q) = \frac{0.01506q+0.09996}{q^2-0.6481q-0.237}$$, of which the frequency function can be found in figure 5.11.

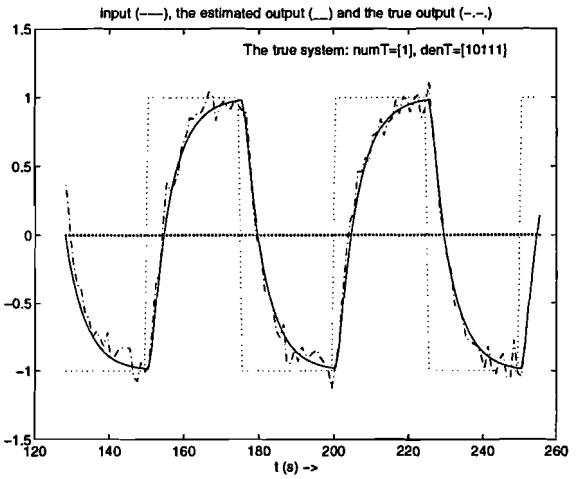

figure 5.10 The input and output signals

figure 5.11: The frequency function of the system and of the model

As can be seen in figure 5.11 the model is (still) very good at the signal frequency of 0.13 rad/s and so the estimated output matches the true output very well in figure 5.10.

Again the model error is computed with both methods of Goodwin. Now noise is present which is not incorporated in the algorithm with the PDF for the frequency function of the undermodelling. The results for this algorithm only showed an exponentially decreasing function of the frequency (the error becomes 0 when $N \rightarrow \infty$) with a vertical asymptote for $\omega=0$ and are not printed. Estimation of the variance parameters of the impulse response of the undermodelling resulted in $\hat{\alpha} = 0.1171$ and $\hat{\lambda} = 0.9078$ and the variance of the noise $\hat{\sigma}_v^2 = 0.0058$. The corresponding model error is plotted in figure 5.12. This looks interesting: more dynamics are visible, there is a minimum for the signal frequency ω_0 (and for $3\omega_0$) and the

[1] For simulation the Matlab function 'dlsim' has been used. This function computes analytical (exact) output samples with a time spacing t_δ of a system with a certain input. In the implementation, the time spacing t_δ also is the sample period of the identification and error algorithms and so t_δ has to be chosen such that aliasing is avoided.

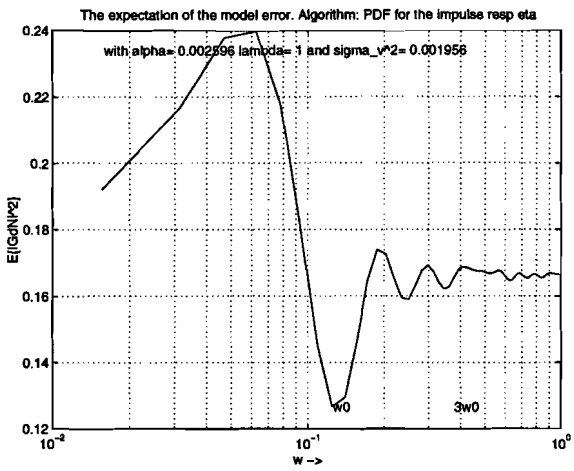

figure 5.12 The model error according to the error algorithm with a PDF for the frequency function

plot looks like the one printed in [Goodwin 92]. However these results are obtained with undersampling. When the system is simulated at an appropriate frequency, then the dynamics disappear and only the two asymptotes are visible and the algorithm became 'useless' again. We should conclude that the dynamics visible in the example of Goodwin result from aliasing! These results are not very promising for the next paragraph where we try to use the algorithms with a practical and so more complicated process.

5.3 A practical process

example 5.4

Since a long time the Measurement & Control group is especially interested in the 'ball balancing process'. This process can serve as a good example of what measurement & control can do.

In the group there is a construction of the 'ball balancing process' available. This process has been identified with the stochastic identification algorithms. With a sample frequency of 100Hz, the resulting 4-th order ARX model for the ball position x as a function of the actuator voltage has two complex conjugate poles just outside the unit circle and so the system is not stable. In order to make it usable with the error algorithm the unstable poles are slightly moved to make the system stable:

$$\frac{x}{V} = \frac{-9.94670*10^{-6}z^{-1} - 9.39704*10^{-5}z^{-2} - 7.96337*10^{-5}z^{-3} - 6.03207*10^{-6}z^{-4}}{1 - 3.4269z^{-1} + 4.2847z^{-2} - 2.2886z^{-3} + 0.4309z^{-4}}.$$

This is (still) quit a troublesome system, as can be seen from the poles:

$p_1 = 0.96502 + 0.01913j$, $p_3 = 0.99999$,

$p_2 = 0.96502 + 0.01913j$, $p_4 = 0.43387$.

These poles with their corresponding frequencies of 3.6, $1.0 \bullet 10^{-3}$ and 83.5Hz specify the frequency region of interest. (Here $z = e^{-st_\delta} \Leftrightarrow s = -f_s \ln(z)$ has been used.) This is a very broad range of frequencies so this makes it a difficult process to identify.

Let's see what happens if we try to identify this process according to the rules for experiment design of appendix C. The sample frequency should be something like 30 times the process bandwidth. The bandwidth can be found from the discrete Bodediagram, which can be constructed with the Matlab function 'dbode' and in this function the sample period 0.01s has to be used. The static gain of the system is 2100 = 66.5dB and the -3dB point is situated at $\omega = 0.01$ rad/s so this system has a very small bandwidth $\omega_B = 0.01$ rad/s = 0.0016Hz. Then the sample frequency for identification would become $f_{s,i} \approx 0.05$Hz, but this is much too slow to capture the dynamics of the pole in 0.43387. When this pole is taken as a measure for the sample frequency then $f_{s,i} = 3 \bullet 83.5 \approx 250$Hz ($t_\delta \approx 0.004$s). In order to capture all dynamics of the system in the measurements, the experiment duration T should at least be 3 times the maximum time constant of the system so $T = 3 \bullet 1/10^{-3} \approx 3000$s. With this T and t_δ the (minimal) number of samples N is determined and is not less than N = 750,000. This is not a practical number. Note: the kind of a priori information which has been used here is usually not available (in this form) in practical situations.

A second order ARX model is identified (NN=[2, 2 ,0]). This is done at a frequency $f_{s,i}$ = 6.25Hz ($t_\delta = 0.16$) so a (second order) anti-aliasing filter 1/F(s) has to be used with a cut off frequency of about 1Hz: $F(s) = (s+1)^2 = s^2+2s+1$. Take N=10,000 samples, a ZMWN input

with $\sigma_v^2 = 1$ is used and the output is corrupted with ZMWN with $\sigma_v^2 = 0.005$. The resulting model is

$$\hat{G}_N(q) = \frac{0.336815q - 0.443892}{q^2 - 1.993559q + 0.993561}.$$

This transfer function is shown in figure 5.13 together with the true system. If we compare the frequency functions of the 4-th order system and the second order model then we see that the frequency fit is 'perfect' at two frequencies.

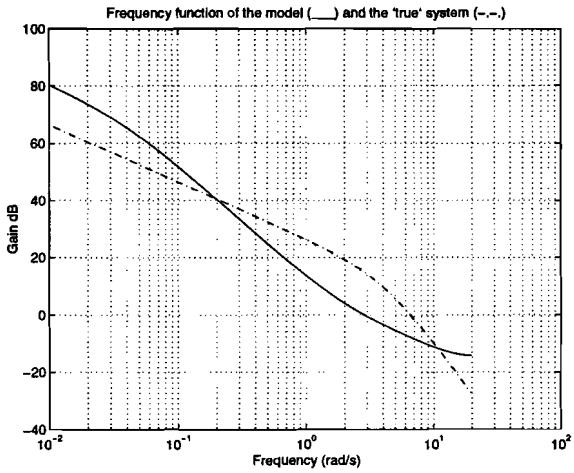

figure 5.13: The Frequency Function of the estimated model and of the true system

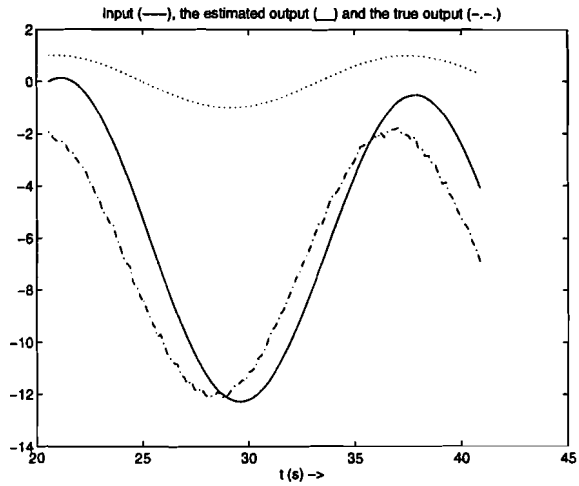

figure 5.14: A sinusoidal input signal and the corresponding true output and the estimated output

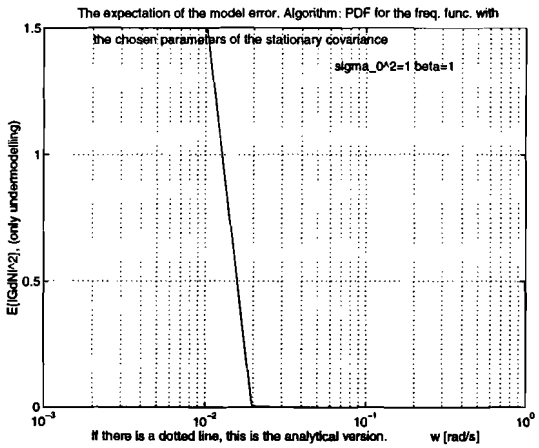

figure 5.15: The undermodelling error according to the algorithm with a PDF for the Frequency Function

For experimental data for the error algorithm 1000 samples of a sinusoidal input signal are used with amplitude 1 and frequency 0.03Hz. The output is disturbed with ZMWN with variance 0.01. This data is entered in the error algorithm.

The maximum frequency which is considered is 1 rad/s ≈ 0.16Hz. N = 128 is used. The time interval which is considered has length $N \bullet t_\delta \approx 20.5$s

The expectation of the (undermodelling) error in case of GOODWIN1 (a PDF for the Frequency Function of the undermodelling) is plotted in figure 5.15 for $\sigma_0^2 = 1$ and $\beta = 1$.

The log likelihood function which has to be maximized in case of GOODWIN2 (PDF for the impulse response of the undermodelling) is

shown in figure 5.16. This is not a complicated function and so the estimates are found quite quick.

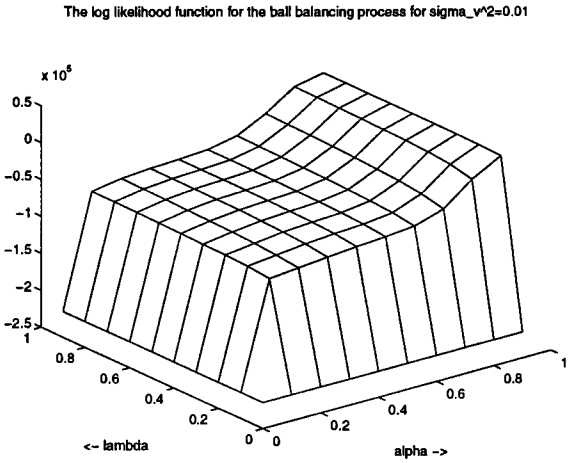

figure 5.16: The log likelihood function for the ball balancing process with a sinusoidal input and output noise variance 0.01

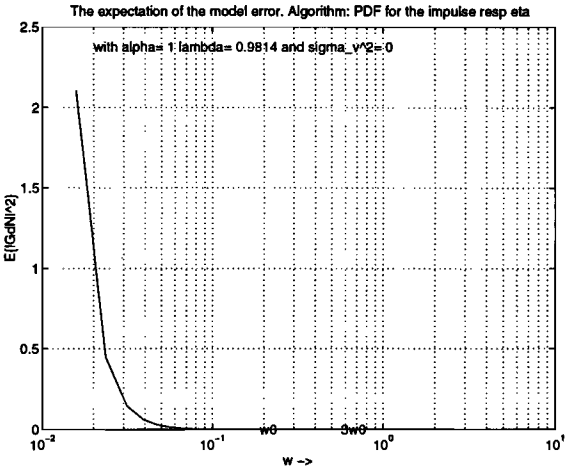

figure 5.17: The expectation of the model error according to the algorithm with a PDF for η

The estimated covariance parameters for the undermodelling are $\hat{\alpha} = 1$ and $\hat{\lambda} = 0.98$. The estimated noise variance $\hat{\sigma}_v^2 = 0$ and this shows that this parameter of the noise cannot be estimated in general. The resulting estimation of the model error (which now also only contains the undermodelling error) is plotted in figure 5.17. It can be seen that both versions of the undermodelling error are in agreement with each other, but again the error is not in agreement with the expectation we have, looking at the Frequency Function. As expected, there is an asymptote at the model error axe (for $\omega = 0$) and one at the frequency axe ($\omega \rightarrow \infty$).

From all the examples it is clear that in case of an ARX model *both* algorithms result in very coarse approximations of the estimation of the model error, which are of little use in practice. The asymptotes seem to be so dominant that there is no place left for more subtle variations in the model error, which could agree with the Frequency Functions.

6. Notation and definitions for a deterministic embedding

6.1 About deterministic identification

If in a method of system identification the system is considered to be deterministic and also the noise is represented by a deterministic quantity, then this method is called deterministic. A type of model which can easily be obtained through a deterministic approach is the *Frequency Function*. The Frequency Function $G(e^{j\omega})$ tells what happens with a sinusoidal input: the output is a sine with the same frequency, the amplitude is multiplied by $|G(e^{j\omega})|$ and the phase is shifted by $\arg\{G(e^{j\omega})\}$. The Frequency Function is a so called *nonparametric* model representation.
A simple way of frequency response analysis is *sine-wave testing*: apply sine waves with specific frequencies to the system and measure the amplitude and the phase-shift of the resulting output. This method has several disadvantages. The transient term can not be identified, but will have strong influence on the model if it has not died out. Since for each frequency a sine wave has to be applied and one has to wait until the transient died out, this method is very time-consuming. Sine-wave testing with correlation suppresses the noise as is explained in [Stochastische Systeemtheorie].

Fourier analysis is another and faster way of frequency response analysis and the resulting model is called the *Empirical Transfer Function Estimation* (ETFE, see [Ljung 87]):

$$\mathring{G}_N(e^{j\omega_l}) = \frac{Y_N(\omega_l)}{U_N(\omega_l)} \qquad \omega_l = \frac{2\pi l}{N} \text{ for } l = 1,....,N$$ with $Y_N(\omega)$ and $U_N(\omega)$ the DFT's of

the input and output. The circle above G denotes that the ETFE is a very rough estimate of the transfer function. This can be explained from the fact that a data series of N input and N output samples is reduced to N/2 complex numbers. The ETFE is the true frequency response plus a bias and a (zero mean) noise term. The ETFE is of increasingly good quality, if a periodic input is used and the number of samples increases. If the input is not periodic, the estimate does not improve as the number of samples increases because also the number of parameters increases. The ETFE can be improved by using a *smoothing* frequency window, which averages the frequency function over a frequency interval. This works because for every system there should be some correlation between two subsequent frequency points.

Smoothing introduces or enlarges the bias and reduces the variance term in the ETFE. See [Stochastische Systeemtheorie].

Since the beginning of the nineties one started to develop (deterministic) identification methods which are strongly related to the environment of Robust / H_∞ control. Norms, like H_∞, play an important role in this and define the largeness of signals and transfer functions. (See appendix A for the different norms.) In Robust control techniques the plant uncertainty is measured in the frequency domain in terms of an ∞-norm ball about the nominal model and guaranteed explicit bounds on the H_∞-norm of the identification error are desired. If these bounds are available, a robust controller can, at least in principle, be found which achieves the desired performance. In a H_∞ sense, the goal of identification is to find algorithms which map the experimental data into an identified model for which the worst case identification error converges to zero in the H_∞-norm as the noise goes to zero and the number of data points goes to infinity. Now first the mathematical environment of this Robust identification method is introduced.

6.2 The mathematical embedding of a H_∞ deterministic approach

6.2.1 Hardy space

D_ρ: open disk of radius ρ (centred at the origin in the complex plane)

$D_\rho := \{z \in \mathbb{C} : |z| < \rho\}$

D: open unit disk, in the cases where $\rho = 1$ the 1 is suppressed in the notation.

T_ρ or ∂D_ρ: boundary of D_ρ, a circle of radius ρ $\quad T_\rho = \partial D_\rho := \{z \in \mathbb{C} : |z| = \rho\}$

$\mathcal{H}(D_\rho,M) = \mathcal{H}_{M,\rho}$: Hardy space of bounded (amplitude < M) analytic, discrete, complex functions in the open disc D_ρ. The arguments of the functions lie in the grey plane of figure 6.1 and the two planes, bounded by the striped lines, denote the amplitude bounding of F(z).

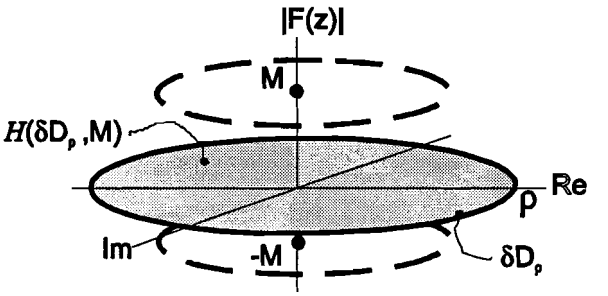

figure 6.1: A visualisation of the Hardy space

$\mathcal{H}_{M,\rho} := \{f: \partial D_\rho \to \mathbb{C} \mid f \text{ analytic in } D_\rho \text{ and } \|F\|_{M,\rho} := \sup_{z \in D\rho} |F(z)| < M\}$

where F(z) is the standard Z-transform of the function f(k) evaluated at 1/z.

Often $M = \infty$: $\mathcal{H}(D_\rho,\infty) = \mathcal{H}_{\infty,\rho}$.

$\mathcal{H}_+$: the set of Hardy spaces with $\rho > 1$, a linear space which is a subset of $\mathcal{H}_\infty$

$$\mathcal{H}_+ := \bigcup_{\rho>1} \mathcal{H}_{\infty,\rho} \subset \mathcal{H}_\infty.$$

A: the disk algebra A, a subset of $\mathcal{H}_\infty$ $\quad A := \{f \in \mathcal{H}_\infty, f \text{ is continuous on unit circle}\}$

$\mathcal{P}$: fixed subset of $\mathcal{H}_\infty$ containing the zero element.

$l_{\infty,\rho}$: a normed space of bounded sequences of positive integers Z_+ in time domain

$\{f : Z_{+,0} \to \mathbb{C} \mid \|f\|_{\infty,\rho} := \sup_{k \in Z_{+,0}} |f(k)|\rho^k < \infty\}$

$L_{\infty,\rho}$: a normed space in discrete frequency domain

$\{f : \partial D_\rho \to \mathbb{C} \mid f \text{ is measurable and } \|f\|_{\infty,\rho} := \sup_{z \in \partial D_\rho} |F(z)| < \infty\}$

$C^k(\partial D_\rho) := \{f: \partial D_\rho \to \mathbb{C} \mid f \text{ is k times continuously differentiable on } \partial D_\rho\} \subset L_\infty$

Note: any $f \in \mathcal{H}_+(D)$ is an element of $C^\infty(\partial D)$ and $\|f\|_\infty = \sup_{z \in D} |F(z)| = \sup_{z \in \partial D} |F(z)|$

6.2.2 Operators

T_n: the truncation operator T_n: $l_\infty \to l_\infty$ such that $\quad (T_n f)_k := f_k \quad$ for k = 0, 1,..., n-1

$(T_n f)_k := 0 \quad$ for $k \geq n$

U_n: the DFT-operator (the discrete and truncated version of the Fourier transform), which transforms the time-domain signals to the discrete frequency domain.

$$U_n: H_+ \rightarrow T_n \; l_\infty \text{ such that } \quad (U_n f)_k := f(e^{2\pi jk/n}) \quad \text{for } k = 0, 1, \ldots, n-1$$
$$(U_n f)_k := 0 \quad \text{for } k \geq n$$

$\bar{B}X(r)$: the ball of radius r for any of the normed spaces X

$$\bar{B}X(r) := \{x \in X : \|x\|_X \leq r\}.$$

Note: the symbol $\|.\|_\infty$ is used in 3 different ways

for $f \in \mathbb{C}^n$ $\quad \|f\|_\infty = \max_k |f_k|$

for $f \in L_\infty$ $\quad \|f\|_\infty = \sup_{z \in \partial D} |F(z)| = \sup_{\theta \in [0,2\pi]} |F(e^{j\theta})|$

for $f \in \mathfrak{H}_\infty$ $\quad \|f\|_\infty = \sup_{z \in D} |F(z)|$.

Each of these uses should be clear from the context.

6.2.3 Polynomials

P_n: collection of all polynomials in z with degree $\leq n$ (a subset of disc algebra *A*).

p_n: a polynomial of the set P_n.

6.2.4 Definitions

Definition 6.1: *Even symmetric*

A function $f_{k,n}$ is even symmetric with respect to k if

(i) $f_{k,n} = f_{-k,n}$ for all k and n i.e. for a fixed n and with k=0 in the origin, both sides of the function are the same;

(ii) $\lim_{n \to \infty} f_{n,n} \log(n) = 0$.

Let $\Delta \mathbf{f}_{n,k} = \mathbf{f}_{n,k} - \mathbf{f}_{n,k+1}$, the first order difference (related to the first order derivative in the continuous domain) and $\Delta^2 \mathbf{f}_{n,k} = \Delta \mathbf{f}_{n,k} - \Delta \mathbf{f}_{n,k+1}$, the second order difference in discrete time domain.

Definition 6.2: *Convex*

A function $f_{n,k}$ is called convex at k if $\Delta^2 f_{n,k} \geq 0 \; \forall$ n and is called a convex function if $\Delta^2 f_{n,k} \geq 0$ for all k, so starting in a minimum of $f_{n,k}$ this function is increasingly rising for increasing k.

Definition 6.3: *Concave*

The same for $\Delta^2 f_{n,k} \leq 0 \; \forall$ n, so loosely speaking $f_{n,k}$ is an increasingly falling function.

6.3 The identification and control environment

6.3.1 Time response data

A data set of N input and output samples is collected: $E_t^N(H,v)$: $\mathcal{H}_+ \times \ell_\infty \rightarrow T_N\ell_\infty$ where $E_t^N(H,v) := T_N\{(h*u)+v\}$.
h is the impulse response of the system and u is the input. The time data is corrupted with noise v of which the amplitude is bounded by ε_t:
$v_k \in \{ v = (v_1, ..., v_N) \in \mathbb{C}^N: \|v\|_\infty \leq \varepsilon_t \}$.

6.3.2 Frequency response data

The input-output data in time domain of the system is transformed to N_f samples of experimental frequency response data: E_f^{Nf}: $\mathcal{H}_+(D) \times B_{Nf}(\varepsilon_f) \rightarrow \mathbb{C}^{Nf}$ where
$E_{f,k}^{Nf}(H,w) := H(j\omega_k) + w_k$.
$H \in \mathcal{H}_+$ and $\omega_k = 2\pi(k-1)/N_f$ for the discrete time case ($k = 1,2,...,N_f$). The frequency response data is corrupted with (frequency) noise w_k which is bounded in amplitude by ε_f:
$w_k \in \{ w = (w_1, ..., w_{Nf}) \in \mathbb{C}^{Nf}: \|w\|_\infty \leq \varepsilon_f \} := B_{Nf}(\varepsilon_f)$.

6.3.3 Class of systems under consideration

The class of systems to which the unknown system will be assumed to belong is *stable*, *SISO* (Single Input Single Output) possibly *distributed*, *LTI* (Linear Time-Invariant), *causal* and *discrete time*.

The system transfer function is defined by $H(z) = \sum_{k=0}^{\infty} h(k) z^k$

i.e. the standard Z-transform evaluated at 1/z.
These are those elements of $\mathcal{H}_\infty$ that admit analytic continuations to a disk D_ρ ($\rho > 1$).
This corresponds precisely to the class of *exponentially* and *BIBO* (Bounded Input Bounded Output) stable linear time-invariant systems, the set $\mathcal{H}_+$.
The following statements, which characterize the time domain properties of this class, are equivalent:

- $H(z) \in \mathcal{H}_+(D)$;
- there exists $M_0 < \infty$, $\rho_0 > 1$ such that $|h(k)| \leq M_0\rho_0^{-k}$ ($k=0, 1, ...$), the amplitude of the impulse response h(k) is smaller than some decreasing exponential;
- there exists $\rho_0 > 1$ such that $\sum_{k=0}^{\infty} |h(k)|\rho_0^k < \infty$.

From this it follows that ρ (or sometimes σ is used in literature) is a *lower bound on the relative stability of transfer function H*;

$\mathcal{H}(\mathbb{C}_{\sigma+},M):= \{h \in \mathcal{H}(\mathbb{C}_{\sigma+}) : \sup |H(s)| < M\}$ $(s \in \mathbb{C}_{\sigma+})$

ρ is strictly less than ρ_0 so $\sup_{|z|=\rho} |H(z)| = \sup_{z \in D_\rho} |H(z)| < M$

and M is an *upper bound on the worst case steady state gain* over all exponentially weighted complex sinusoidal inputs of the form $u(k) = \rho^{-k}e^{j\omega k}$.
The parameters ρ (or σ) and M correspond to *a priori knowledge* of the unknown system.

Definition 6.4: *Admissible*
A subset $\mathcal{P} \subseteq A$ is called admissible if

1. $\lim_{n\to\infty} \delta_n = 0$ where δ_n is the supremum of the measured data

   $\delta_n := \sup \{E^n(H): H \in \mathcal{P}\} = \sup \{ \inf\{\| H-p_n\|_\infty : p_n \in P_n\} \}$
2. $M_s := \sup \{ \|H\|_\infty : H \in \mathcal{P} \} < \infty$

Roughly speaking, admissiblity requires that the set of systems over *A* of space $\mathcal{H}_\infty$ be bounded and be uniformly approximable by polynomials. It is assumed that $\mathcal{P}$ is an admissible set so it is required that $\mathcal{P}$ is totally bounded in $\mathcal{H}_\infty$. A class of systems with a lowerbound on the relative stability and an upperbound on the steady-state gain is admissible.
Note: - the collection exponentially stable systems $\mathcal{P}:= \mathcal{H}(D_\rho, M)$ is admissible;
- sets having certain 'smoothness' are also admissible.

6.3.4 Problem definition

In the approach of Helmicki, Jacobson and Nett in [Helmicki 90a] and [Helmicki 91] two related control-oriented system identification problems for stable, linear and time-invariant systems are solved. The first problem is identification of the plant frequency response from a noisy, time series output of the plant and finding an expression for the worst case model error. This problem leads naturally to the second problem, which involves identification of the plant transfer function in H_∞ from a finite number of noisy samples of the plant frequency response and again determination of an upperbound of the model error.

I) The experimental data consists of N noisy time samples and is mapped to noisy values of the frequency response of the system at a given set of N_f different frequencies.
The problem of **point frequency response identification** according to Helmicki e.a.:
Assume: that the plant H, whose frequency response point sample $H(e^{j\omega})$, $\omega \in [0,2\pi)$, has to be identified, is an element of $\mathcal{H}_+$ and that the noise v present during identification is bounded by ε_t.
Given: 1) plant a priori information in the form of the pair $(\rho, M) \in (1,\infty] \times [0,\infty)$ for which it is known that $H \in \bar{B}\mathcal{H}_{M,\rho}$; so a known lower bound on the relative stability of transfer function $\rho > 1$ and a known upper bound M on the worst case steady state

gain;
2) time noise a priori information in the form of a bound $\varepsilon_t \in [0,\infty)$ on the level of corruption, i.e. $v \in \bar{B}\ell_\infty(\varepsilon_t)$; the measurements of the output are corrupted by coloured noise v of which the amplitude is bounded by the known value ε_t;
3) for each data a posteriori information level $N \in \mathbb{N}$, the data a posteriori information defined by the experiment operator in time domain: $E_t^N(H,v)$: $\mathcal{H}_+ \times \ell_\infty \rightarrow T_N\ell_\infty$,

where $E_t^N(H,v) := T_N\{(h*u)+v\}$ and $u(k) = \begin{cases} \alpha e^{-j\omega k} & k \geq 0 \\ 0 & k < 0 \end{cases}$

with $\alpha \in (0,\infty)$ a fixed constant and $0 \leq \omega \leq 2\pi$.
The system with transfer function H is excited by N_f different complex exponential input signals (sinusoidals) of N time samples. The signals have N_f different

frequencies ω $E_t^N(H,v) := \left[H\left(e^{i\frac{2\pi(k-1)}{N}}\right)+v_k\right]_{k=1}^{N}$.

Find: a plan of algorithms A_t^N such that for each a posteriori information level N, the algorithm A_t^N: $T_N l_\infty$ maps the given experimental time data into a point frequency response estimate $A_t^N(E_t^N(H,v)) \in \mathbb{C}$)[1] in such a way that the worst case identification error

$$e_t^N(A_t^N;\rho,M,\epsilon_t) := \sup_{\substack{H\in\bar{B}\mathcal{H}_{M,\rho} \\ v\in\bar{B}\ell_\infty(\epsilon_t)}} \left|H(e^{j\omega}) - A_t^N\left(E_t^N(H,v)\right)\right|$$

converges as follows:

$$\lim_{\substack{\epsilon_t\to 0 \\ \rho\to\infty \\ M\to 0}} e_t^N(A_t^N;\rho,M,\epsilon_t) = 0 \quad \textit{and} \quad \lim_{\substack{\epsilon_t\to 0 \\ N\to\infty}} e_t^N(A_t^N;\rho,M,\epsilon_t) = 0.$$

(Any system identification algorithm should be able to identify a system exactly if the information is complete and uncorrupted.)
In addition, derive explicit bounds on $e_t^N(A_t^N;\ \rho,\ M,\ \varepsilon_t)$ as a function of ρ, M, ε_t and N.

II) The problem of **identification in H_∞** formulated by Helmicki e.a. resembles the point frequency response identification problem:

[1] It is important to make a distinction between the algorithm A_t^N for identification and the resulting models $A_t^N(E_t^N(H,v))$. For example, the identification algorithm is independent of the input u, but the model of course depends on the input used.

Assume: that the 'true' unknown system H, whose transfer function has to be identified, is stable, linear, time invariant, discrete-time and an element of $\mathcal{H}_+$ and that the frequency noise w present during the identification process is bounded, i.e. $w \in \ell_\infty$.

Given: 1) plant a priori information in the form of the pair $(\rho, M) \in (1,\infty] \times [0,\infty)$ for which it is known that $H \in \bar{B}\mathcal{H}_{M,\rho}$;

2) frequency noise a priori information in the form of a bound $\varepsilon_f \in [0,\infty)$ on the level of corruption, i.e. $w \in \bar{B}\ell_\infty(\varepsilon_f)$;

3) for each data a posteriori information level N_f (number of corrupted frequency response estimate samples) $\in \mathbb{N}$, the data a posteriori information defined by the frequency experiment operator: $E_f^{Nf}(H,w)$: $\mathcal{H}_+ \times \ell_\infty \rightarrow T_{Nf}\ell_\infty$, where $E_f^{Nf}(H,w) := U_{Nf}H + T_{Nf}w$.

Find: a plan of algorithms A_f^{Nf} such that for each a posteriori information level N_f, the algorithm A_f^{Nf}: $T_{Nf}l_\infty$ maps the given experimental frequency data into a *transfer function estimate*, the *identified model*, $\hat{H}_{id}^{Nf} = A_f^{Nf}\left(E_f^{Nf}(H,w)\right) \in \mathcal{H}_+$ in such a way that the worst case frequency identification error

$$e_f^{N_f}(A_f^{N_f};\rho,M,\epsilon_f) := \sup_{\substack{H \in \bar{B}\mathcal{H}_{M,\rho} \\ w \in \bar{B}\ell_\infty(\epsilon_f)}} \left\| H - A_f^{N_f}\left(E_f^{N_f}(H,w)\right) \right\|_\infty$$

converges as follows:

$$\lim_{\substack{\epsilon_f \to 0 \\ \rho \to \infty \\ M \to 0}} e_f^{N_f}(A_t^{N_f};\rho,M,\epsilon_f) = 0 \quad \textit{and} \quad \lim_{\substack{\epsilon_f \to 0 \\ N_f \to \infty}} e_f^{N_f}(A_t^{N_f};\rho,M,\epsilon_f) = 0.$$

In addition, derive explicit bounds on $e_f^{Nf}(A_f^{Nf};\ \rho,\ M,\ \varepsilon_f)$ as a function of ρ, M, ε_f and N_f.

This problem definition of system identification is in agreement with the description in §6.1.

Definition 6.5: *Convergent*
An algorithm A_t^N or A_f^{Nf} which results in a model which satisfies the conditions for the error mentioned in the problem definitions above is called convergent.

Definition 6.6: *Robustly convergent*
If, in addition to convergence, the algorithm does not depend on the a priori information ρ and M then it is robustly convergent. The algorithm converges even when the available a priori information is incorrect.

Definition 6.7: *Intrinsic error*

$e_{in,t}^{N} = \inf_{A_t^N} e_t^N$, *or* $e_{in,f}^{N_f} = \inf_{A_f^{N_f}} e_f^{N_f}$ i.e. there is always an intrinsic level of uncertainty,

determined by the available information, which establishes fundamental limits on how much uncertainty can be reduced through the use of a particular set of information. This fundamental limit sets lower bounds on the achievable accuracy of any algorithm.

Definition 6.8: *Optimal*
A particular algorithm *A* is *optimal* if for each resulting model and for each choice of ρ, M, ε and N:

$e = e_{in}$, i.e. the (true) model error is equal to the intrinsic error.

Now this environment will be used for the introduction of a class of deterministic identification algorithms in the next chapter.

7. A deterministic approach of system identification

In this chapter a class of modelling techniques will be regarded which are based on the principle of the Frequency Function and ETFE models, but for which *explicit worst case error bounds* are derived. Through these error bounds this deterministic approach of system identification is more appropriately formulated for use in Robust control. In literature this deterministic approach is also called the worst-case approach. It is a *control-oriented system identification* method.

The several algorithms resulting of the problem definition of Helmicki e.a. as described in chapter 6 are characterized by a two-stage structure: the first stage involves transformation of time responses to a frequency response and the second stage involves approximation of the frequency response by some kind of interpolation and then finding the best stable approximation of this first, possibly nonstable, approximation. Creation of the frequency response in the first stage can be done with the ETFE modelling technique (see §6.1) or by sine-wave testing (the way which is described in appendix B of [Helmicki 90a]). Interpolation in the second stage can be performed by taking the inverse Discrete Fourier Transform of the frequency response and multiplication by a suitable window function. A stable approximation can be found by using Nehari's theorem.
The two-stage algorithms are also non-parametric so less a priori information is required then in a parametric case. Notions of intrinsic error, algorithm optimality etc., as specified in chapter 6, can be quantified.
The set of systems to which the unknown system belongs must be admissible (see definition 6.4).

The structure of the two-stage algorithm will be discussed in the first paragraph. In the second paragraph the different forms of the algorithm will be looked at. These algorithms all are nonlinear. In §7.3 something will be said about linear algorithms and in §7.4 a few notes will follow on the implementation of the algorithms in Matlab. Proofs of theorems will be omitted for size reduction of this report, references to the literature are given where the proofs can be found.

7.1 The two-stage structure

Remark: the notation which is used is slightly different from the notation used in [Helmicki 90a] and [Helmicki 91]: like in this whole report the number of data is 'N' and here the number of different frequencies is 'N_f' and not 'n' like in the articles of Helmicki and co-workers.

7.1.1 First stage: *Point frequency response identification*

In case of sine-wave testing a complex sinusoidal test input u at a desired frequency $\omega \in [0,2\pi]$ and with amplitude α is applied to the system as described in chapter 6. The first N values of u and output y, which are both complex signals, are recorded and form the experimental time data $E_t^N(H,v)$. The time output noise v which is present, is bounded in amplitude by ε_t.

Theorem 7.1: A lower bound on the optimal identification error for point frequency response identification is $e_t^N(\rho, M, \varepsilon_t) \geq \min\{M, \varepsilon_t/\alpha\}$.

Note that this error does not depend on the algorithm A_t^N used. This theorem states that the signal-to-noise ratio fundamentally limits identification of the plant frequency response. Proof of this theorem can be found in [Helmicki 91].

Here two plans of algorithms for approximately identifying a plant frequency response will be presented. The first plan attempts to recover the frequency response information contained in the experimental time data $E_t^N(H,v)$, which should consist of complex signals, by processing only the last data point logged:

$$A_t^N\left(E_t^N(H,v)\right) := \frac{\left[E_t^N(H,v)\right]_{N-1}}{\alpha e^{-j\omega(N-1)}} = \frac{y(N-1)}{u(N-1)}.$$ The worst case error properties (i.e. an upperbound) for each of the elements is $$e_t^N\left(A_t^N; \rho, M, \epsilon_t\right) \leq \frac{M\rho}{\rho - 1}\rho^{-N} + \frac{\epsilon_t}{\alpha}.$$

The second plan of algorithms attempts to correlate out the frequency response information through the use of finite length (Inverse Discrete Fourier) transforms:

$$A_t^N\left(E_t^N(H,v)\right) := \frac{1}{\alpha N}\sum_{k=0}^{N-1}\left[E_t^N(H,v)\right]_k e^{j\omega k} = \frac{1}{\alpha N}\sum_{k=0}^{N-1} y(k)e^{j\omega k}.$$ Here the worst case error is

$$e_t^N\left(A_t^N; \rho, M, \epsilon_t\right) \leq \frac{M\rho}{(\rho-1)^2}\frac{1-\rho^{-N}}{N}+\frac{\epsilon_t}{\alpha}.$$

See [Helmicki 91] for the proofs of these expressions. As can be seen, both algorithms are robustly convergent for all admissible values of the parameters α and ω and asymptotically optimal if $\epsilon_t/\alpha < M$. Both algorithms operate linearly on the experimental data.

The worst case errors provide a ball in the complex plane within which the true point frequency response sample is guaranteed to lie. Moreover, given any number of balls for the same frequency point, corresponding to different information levels N (different data lengths) it follows that the true point frequency response sample is guaranteed to lie in the intersection of these balls. Hence a significant reduction in identification error can be obtained in cases where the radius of the smallest ball containing this intersection, is significantly smaller than the radius of any individual ball. One can try to find this ball with a recursive process $E_t^{N-m}(H,v) = T_{N-m}E_t^N(H,v)$.

7.1.2 Second stage: *Identification in H_∞*

The second stage itself can be divided in two steps. In the first step the N_f noisy point samples of the frequency response of the unknown plant are mapped into an L_∞ approximation of the plant. The second step maps the L_∞ approximation into a stable real-rational H_∞ approximation to the unknown plant.
A frequency data record of a posteriori information level N_f can be obtained by solving N_f point frequency response identification problems as described in §7.1.1. The noise w represents bounded perturbations to the true frequency response. The amplitude bound of this frequency noise is ϵ_f. If the experimental response data are obtained via one of the algorithms of §7.1.1 then ϵ_f can be obtained using one of the *worst case time identification errors* $\epsilon_f = e_t^N(A_t^N; \rho, M, \epsilon_t)$.

Theorem 7.2: A lower bound on the optimal identification error for the problem of identification in H_∞ is $e_f^{Nf}(\rho, M, \epsilon_f) \geq \min\{\epsilon_f, M\}$.

A model cannot be identified with an uncertainty better than can be measured at any frequency. Proof of this theorem can be found in [Helmicki 91].

Step 1: L_∞ approximation of the plant
The idea is to interpolate the N_f samples of the experimental frequency response data $E_f^{Nf}(H,w)$ with an interpolation operator $\mathbb{I}$: $\mathbb{I}_{Nf}\{E_f^{Nf}(H,w)\}$. The interpolating Fourier series coefficients h_k of $E_f^{Nf}(H,w)$ are an L_∞ approximation to H. However in this discrete algorithm only DFT coefficients $\hat{h}_k$ of $U_{Nf}\{E_f^{Nf}(H,w)\}$ can be computed. Fortunately also truncations of the interpolating Fourier series provide an L_∞ approximation to H.

Theorem 7.3

$$F_{N_f}\left\{E_f^{N_f}(H,w)\right\}(e^{j\theta}) := \sum_{k=-n}^{n} h_k \mathbb{I}_{N_f}\left\{E_f^{N_f}(H,w)\right\} e^{jk\theta} = \sum_{k=-n}^{n} w_{n,k}\tilde{h}_k E_f^{N_f}(H,w)\, e^{jk\theta}$$

where $w_{n,k}$ is a frequency *window function* ($-n \le k \le n$) and n is a free parameter which specifies how many Fourier series coefficients will be used in the truncation. Let n be any given monotone decreasing function of the number of data samples N_f such that

$$\lim_{N_f \to \infty} \frac{N_f^2}{n(N_f)} = 0 \ .$$

From this theorem it follows that for each $k \in Z$ $\mathbf{h_k \bullet \mathbb{I}_{Nf}\{U_{Nf}\{E_f^{Nf}\}\} = w_{n,k} \hbar_k U_{Nf}\{E_f^{Nf}\}}$.
The proof of this can be found in [Helmicki 91].

So the first step of the second stage comes down on the calculation of the discrete Fourier transform coefficients of the frequency response estimates E_f^{Nf}, i.e. the N-point IDFT of the experimental frequency data:

$$\tilde{h}^{N_f}(k) = \frac{1}{N_f}\sum_{l=0}^{N_f-1}\left[E_f^{N_f}(H,w)\right]_{l+1} e^{-lkj\frac{2\pi}{N_f}} \quad \text{where } 0 \le k < N_f.\ \hbar^{Nf} \text{ is assumed to be periodical}$$

so $$\tilde{h}^{N_f}(k)\Big|_{k=0}^{N_f} = \tilde{h}^{N_f}(k)\Big|_{k=-N_f}^{0} .$$

And then attenuation of the DFT coefficients using a suitable window function $w_{n,k}$. The

preidentified model $\hat{H}_{pi}^{n}$ becomes: $$\hat{H}_{pi}^{n}(z) = \sum_{k=-n}^{n} w_n(k)\tilde{h}^{N_f}(k) z^{k}.$$

The L_∞ approximation is a good, but possibly unstable, approximation to the given frequency response data. It is not necessarily causal as it may have nonzero negative Fourier coefficients. The L_∞ approximation is only guaranteed to converge on the unit circle and an identified model has to agree with the true unknown system over the whole unit disk, i.e. a H_∞ approximation is wanted.

The window function allows the Fourier series coefficients to be calculated directly from the frequency response estimates and controls the effects of the noise in the samples as will be shown in §7.2. For each choice of the window function, a different identification algorithm results. The window function determines completely the properties of the second stage of the algorithm. The choice of the free parameter n determines the order of the preidentified model.

Step 2: A stable real-rational H_∞ approximation to the unknown plant
The problem is to find the best analytic approximation to the preidentified model and *Nehari's theorem* can be used for this. The first step approximation is further approximated to obtain a stable, identified model.
The *identified model* $\hat{H}^n_{id}$ is taken as: $\hat{H}^n_{id}(z) := \text{argmin} \{\|\hat{H}^n_{pi} - H\|_\infty : H \in \mathcal{H}_\infty\}$.

This second stage is also treated in [Gu 92]. Pay attention to the different interpretation of the 'two stages' of the algorithm as described in [Gu 92]. Here they call the two *steps* of the second stage of the algorithm of Helmicki the two stages on which they base the name 'two-stage algorithm'.

7.1.2 Convergence

Lemma 7.1a: The second step and so the second stage of the two-stage algorithm is *robustly convergent* if step 1 of the second stage is robustly convergent.

Lemma 7.1b: Step 1 of the second stage is *robustly convergent* if the identification error in

step 1 $e^{Nf}_{pi} := \sup \{\| H - \hat{H}^n_{pi}\|_\infty : w \in B_{Nf}(\varepsilon_f), H \in \mathcal{P}\}$ satisfies $\lim_{\substack{\epsilon_f \to 0 \\ N_f \to \infty}} e^{N_f}_{pi} = 0$.

Remember $\mathcal{P}$ is a fixed subset of $\mathcal{H}_\infty$ containing the zero element, the set $\mathcal{P}$ captures the prior information on the system. Then the final identification error $\mathbf{e^{Nf} \leq 2e^{Nf}_{pi}}$ and the two-stage algorithm is robustly convergent.
The two-stage algorithms only operate on the available a posteriori information so they can be referred to as being *untuned*. The algorithms are totally independent of the available a priori information. These robustly convergent plans of algorithms are guaranteed to converge even in cases where the available a priori information is incorrect.

Theorem 7.4: Robust convergence of step 1 of stage 2 in terms of the properties of the window function.
Suppose that the window function $w_n(k)$ is even symmetric with respect to k (see definition 6.1). Then step 1 is robustly convergent if the window function satisfies:

(i) $\lim_{n\to\infty} \Delta^2 w_n(k) = 0$ for k = 0,1,..... with $\Delta^2 w_n(k)$ as defined in chapter 6;

(ii) $N_s := \text{limsup} \left\{ N_n := n|\Delta w_n(n-1)| + \sum_{k=0}^{n-2} (k+1)|\Delta^2 w_n(k)| : n \geq 0 \right\} < \infty$

(iii) $\lim_{n\to\infty} w_n(0) = 1$.

Consequently, if the above conditions hold, then the two-stage nonlinear identification algorithm is robustly convergent. See [Gu 92] for more about this theorem. The idea of sampling twice differentiable functions to arrive at a window function is quite common in digital filtering literature.

7.1.3 Worst case approximation error

To analyze the identification error e_{pi}^{Nf} in step 1 of the second stage, note that the N_f point IDFT of the frequency data can be written as $\tilde{h}^{Nf}(k) = H^{Nf}(k)+w^{Nf}(k)$ and therefore the preidentified model can be written as $\hat{H}_{pi}^{n} = H_{pi}^{n}+W_{pi}^{n}$.
The worst case identification error at the first step satisfies

$$\sup \left\{ \|\hat{H}_{pi}^{n} - H\|_{\infty} : w \in B_{Nf}(\varepsilon_f), H \in \mathcal{P} \right\} \leq$$
$$\sup \left\{ \|\hat{H}_{pi}^{n} - H\|_{\infty} : H \in \mathcal{P} \right\} + \sup \left\{ \|W_{pi}^{n}\|_{\infty} : w \in B_{Nf}(\varepsilon_f) \right\} \qquad (7.1)$$

The first term on the right hand side of (7.1) is the *worst case undermodelling* (approximation) error:

$e_{Nf,n}^{under}(\rho, M) = \sup \{ \|\hat{H}_{pi}^{n} - H\|_{\infty} : H \in \mathcal{P} \}$ corresponds to the noise free case.

The second term on the right hand side of (7.1) is the *worst case noise / variance error*:

$e_{Nf,n}^{noise}(\varepsilon_f) = \sup \{ \|W_{pi}^{n}\|_{\infty} : w \in B_{Nf}(\varepsilon_f) \}$ which corresponds to pure noise case (H=0).

In a theorem in [Gu 92] some bounds for the noise error are given for cases when the window function is symmetric and convex or concave (see definitions 6.2 and 6.3). In general it is not possible to reduce the undermodelling error and noise error simultaneously and there is a trade-off between noise reduction and system approximation. In §7.2.2 a *parametrized window* is introduced which incorporates this trade-off as a design parameter.

7.2 Specific forms of the second stage of the two-stage algorithm

The choice of the window function $w_n(k)$ is crucial and determines completely the properties of the second stage of the resulting identification algorithm. The system identification problem is reduced to the optimal design of the window function.

7.2.1 Based on spline interpolation: Helmicki, Jacobson and Nett

The approach with spline interpolation is according to the articles [Helmicki 90a] and [Helmicki 91]. Spline interpolation can be used to obtain an L_∞ approximation. This approximation is an estimation of the system H for the discrete frequency samples of the experimental frequency response data $E_f^{Nf}(H,w)$.

Lemma 7.2: Spline interpolation.
Let $f \in \ell_\infty$ and $\|f\|_\infty = 1$. The *linear* spline $S_{Nf}\{f\} \in L_\infty$ interpolates the first N_f components of f at the points $z_k = e^{j\theta_k}$ with $\theta_k = 2\pi k/N_f$ for $k = 0,...,N_f-1$:

$$S_{N_f}\{f\} = \begin{cases} f_k + (\theta-\theta_k)\left(\dfrac{f_k - f_{k+1}}{\theta_k - \theta_{k+1}}\right) & \theta_k \le \theta < \theta_{k+1},\ k = 0,...,N_f-2 \\ f_{N_f-1} + (\theta-\theta_{N_f-1})\left(\dfrac{f_{N_f-1} - f_0}{\theta_{N_f-1} - 2\pi}\right) & \theta_{N_f-1} \le \theta < 2\pi \end{cases}$$

The spline interpolant is the value of the function f at sample k plus some displacement $\theta-\theta_k = \delta\theta$ times the slope between the two successive samples and so this is indeed an interpolation. The linear spline interpolant is a first order polynomial interpolant.

- The spline operator S_{Nf} is linear and the induced norm is $\|S_{N_f}\| := \sup_{\|f\|_\infty=1} \|S_{N_f}\{f\}\|_\infty = 1$ since

f_k appears linearly in the expression for the spline operator and $\max_{k=0,...,N_f-1} \{f_k\} \le 1$.

- If $f \in C^1(\partial D)$ then $\|f - S_{Nf}\{f_s\}\|_\infty \le (4\pi/N_f)\|f'\|_\infty$, where f_s is a section of the samples of the function f and f' denotes the first derivative of f.
If in addition $f \in C^2(\partial D)$ then $\|f - S_{Nf}\{f_s\}\|_\infty \le (\pi/N_f)^2(\|f'\|_\infty + \|f''\|_\infty)$.

Fact: If $f \in \mathcal{H}_+$, then

$$\|f - S_{N_f} U_{N_f}\{f\}\|_\infty \le \min\left\{\frac{4\pi}{N_f}\|f'\|_\infty, \left(\frac{\pi}{N_f}\right)^2 (\|f'\|_\infty + \|f''\|_\infty)\right\}.$$

Fast dynamics in a function f (so large norms of the slope) result in a larger upperbound of the difference (error) between the function f and it's truncated spline interpolant $S_{Nf}\{f\}$. This is also the case when the functional values are corrupted with high frequency noise. Since the induced norm is smaller than 1 is the coefficient of the ε_f term also always smaller than 1, correponding to noise reduction. This establishes the capability of linear spline interpolants to control the effects of noise in the interpolated data.

- Let $\rho > 1$ and $M < \infty$. If $H \in \mathcal{H}(D_\rho, M)$, then for the k-th derivative of h

$$\|H^{(k)}\|_\infty \le \frac{Mk!}{(\rho-1)^k} .$$

This characterizes the approximation power of linear splines in case of noise-free function samples.

First step: The L_∞ approximation
The first step has been described in §7.1.
* Calculate the Discrete Fourier Transform coefficients of the experimental data $E_f^{Nf}(H,w)$:

$$\tilde{c}_k := \frac{1}{N_f} \sum_{m=1}^{N_f} \left[H\left(e^{j\frac{2\pi(m-1)}{N_f}}\right) + w_m \right] e^{-jk\frac{2\pi(m-1)}{N_f}} \qquad (-N_f \le k \le N_f)$$

* Choose the free parameter n so that $\lim_{N_f \to \infty} \frac{N_f^2}{n(N_f)} = 0$. For example take $n = N_f^3$.

Attenuate the DFT-coefficients with the *sinusoidal window* to obtain the spline Fourier series coefficients.

$$w_n(k) = \begin{cases} 1 & k = 0 \\ \left(\frac{n}{k\pi}\sin\left(\frac{k\pi}{n}\right)\right)^2 & -n \le k \le n \end{cases}$$

Then the preidentified model is:

$$\hat{H}_{pi}^{n}(z) = \sum_{k=-n}^{n} w_n(k) c_k z^k = F_n\left\{E_f^{N_f}(H,w)\right\}$$

which is the truncated spline Fourier series.

Lemma 7.3a: The distance between the true transfer function H(z) and the linear spline interpolant $S\{E_f^{Nf}(H,w)\}$ satisfies:

$$\left\| H - S\{E_f^{N_f}(H,w)\} \right\|_\infty \le \min\left\{ \frac{4M\pi}{(\rho-1)N_f}, \frac{M\pi^2(\rho+1)}{(\rho-1)^2 N_f^2} \right\} + \epsilon_f$$

with $\rho > 1$, $M < \infty$, $\varepsilon_f \ge 0$ and $H \in \mathcal{H}(D_\rho, M)$.

Note that in the error bound the terms due to the level of partialness N_f and the frequency error due to the level of corruprtion ε_f are decoupled. As $N_f \rightarrow \infty$ the linear spline approximation converges within a constant tolerance ε_f. This is a direct consequence of the fact that the induced norm of the spline operator is independent of the number of interpolation points N_f (see lemma 7.2). This property is not shared by other interpolation algorithms and this is the motivation for using linear splines.

Lemma 7.3b: The distance between a complete linear spline interpolant $S\{E_f^{Nf}(H,w)\}$ and the preidentified model $\hat{H}_{pi}^{N}$ (the truncated spline interpolant) satisfies:

$$\left\| S\{E_f^{N_f}(H,w)\} - \hat{H}_{pi}^{n} \right\|_\infty \le \frac{2(M+\epsilon_f)N_f^2}{n\pi^2} \quad \text{with } \rho > 1,\ M < \infty,\ \varepsilon_f \ge 0 \text{ and } H \in \mathcal{H}(D_\rho, M).$$

So by selecting a large enough value for n, an arbitrarily accurate L_∞ approximation can be obtained.

The error bound for the preidentified model $\hat{H}_{pi}^{n}$ is

$$\left\| H - H_{pi}^{n} \right\|_{\infty} \leq \min \left\{ \frac{4M\pi}{(\rho-1)N_f} , \frac{M\pi^2(\rho+1)}{(\rho-1)^2 N_f^2} \right\} + \frac{2(M+\epsilon_f)N_f^2}{n\pi^2} + \epsilon_f \qquad (7.2)$$

Second step: AAK approximation to extract a stable, real-rational and causal H_∞ function from the L_∞ approximation.
In paragraph 7.1 the identified model has been defined as:

$$\hat{H}_{id}^{n} := \textit{argmin} \{ \| \hat{H}_{pi}^{n} - H \|_{\infty} : H \in \mathcal{H}_{\infty}(\mathbb{C}_{+}) \}.$$

To get a causal model, the negative Fourier series coefficients have to be eliminated. The final model is the preidentified model with a transfer function added which compensates for the noncausal part:

$$\hat{H}_{id}^{n}(z) := \sum_{k=-n}^{n} c_k z^k + G(z).$$

This can be done in two ways.

A) Compensate the noncausal part of the preidentified FIR model with an ARX model with exactly the same impulse response for $n \leq k < 0$.

Theorem 7.5: Let $M < \infty$, $\rho > 1$ and $\epsilon_f \geq 0$. For any $H \in \mathcal{H}(D_\rho, M)$ define

$$G(z) = -\frac{\bar{\sigma} \sum_{k=0}^{n-1} U_{n-k} z^k}{z^n \sum_{k=0}^{n-1} V_{k+1} z^k}$$

where $\bar{\sigma}$ is the maximal eigenvalue and $V = [V_1\ V_2\ ..V_n]^T$ and $U = [U_1\ U_2\ ...\ U_n]^T$ the right and left maximal eigenvectors respectively of the Hankel matrix is formed by the negative Fourier series coefficients of the preidentified model:

$$H = \begin{bmatrix} c_{-1} & c_{-2} & \cdots & c_{-n} \\ c_{-2} & c_{-3} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ c_{-n} & c_0 & \cdots & 0 \end{bmatrix}.$$

The term z^n in the denominator is a term z^{-n} in the numerator and accomplishes a shift of the causal ARX model to the negative/noncausal part of the time axis where it has to compensate the preidentified FIR model.

By this AAK approximation, the error bound becomes twice as large as (7.2).

Unfortunately, this expression for G(z) involves exact cancellation of the negative terms $\sum_{k=-n}^{-1} c_k z^k$ and this is numerically not possible. This will be a computational stumbling block.

B) The transfer function G(z) of course can be written as the Z-transformation of the impulse response g(k) of the model which is available, so then the identified model becomes

$$\hat{H}_{id}^n = \sum_{k=-n}^{n} c_k z^k + \sum_{k=-n}^{\infty} g(k+n) z^k = \underline{\sum_{k=-n}^{-1} c_k z^k} + \sum_{k=0}^{n} c_k z^k + \underline{\sum_{-n}^{-1} g(k+n) z^k} + \sum_{k=0}^{\infty} g(k+n) z^k = \sum_{k=0}^{n} c_k z^k + \sum_{k=0}^{\infty} g(k+n) z^k .$$

Now the infinite impulse response g(k) has to be available, but a FIR approximation to the model can be used: $\hat{H}_{id}^n = \sum_{k=0}^{n} c_k z^k + \sum_{k=0}^{n} g(k+n) z^k .$

The resulting FIR approximation may be of extremely high order. The special FIR structure can be exploited to obtain a reduced order model fit, truncating the states of a near balanced realization. (See [Helmicki 90a].)
An easy solution for the implementation of the second stage of the algorithm in Matlab is to make the first n samples of the impulse response of the preidentified model 0 (see §7.4).

The truncation causes an additional error.

Lemma 7.4: The error caused by the truncation is equal to $\left\| G(z) - T_n\{G(z)\} \right\|_\infty \le \frac{M}{\rho^n(\rho - 1)}$

where T_n is the truncation operator as defined in chapter 6.

So the error for the identified model becomes:

$$e^{N_f} = \sup_{\substack{H \in \mathcal{H}(D_\rho, M) \\ w \in B_{N_f}(\epsilon_f)}} \left\| H - \hat{H}_{id}^{n} \right\|_\infty \leq$$

$$2\min\left\{ \frac{4M\pi}{(\rho-1)N_f}, \frac{M\pi^2(\rho+1)}{(\rho-1)^2 N_f^2} \right\} + \frac{2(M+\epsilon_f)N_f^2}{n\pi^2} + 2\epsilon_f + \frac{M}{\rho^n(\rho-1)}$$

If the free parameter n is chosen as $n(N_f) = N_f^3$, which satisfies the condition in §7.1.2, the model error obeys the relation $e^{Nf} = 2\varepsilon_f + O(1/N_f)$. For such choices of n, the algorithm is actually asymptotically optimal to within a factor of 2.
As can be seen, the algorithm is indeed robustly convergent (untuned) and is a nonlinear function of the frequency response information E_f^{Nf}. With the choice of parameter n of above, the order of the identified model is N_f^3 and this is in general very large.

7.2.2 Other window functions

As expressed in the previous sections, by choosing other window functions, new identification algorithms appear. Below a few window functions are shown which are in use in literature and which could be implemented.

Cosine window: window for *second Bernstein procedure*

$$w_n(k) = \cos\left(\frac{k\pi}{2n+1}\right) \qquad (-n \leq k \leq n).$$

Using the theorem of [Gu 92] which specifies the worst case identification error as a function of the character of the window function, the corresponding error for the cosine window is:

$$e^{N_f} \leq \frac{2M\rho}{\rho-1}\left(2\rho^{-n} + \frac{(\rho+1)\pi^2}{2(2n+1)^2(\rho-1)^2}\right)$$
$$+ 2\left(2n\cos\left(\frac{(n-1)\pi}{2n+1}\right) - 1 + C_1\cos\left(\frac{n\pi}{2n+1}\right)\log(n)\right)\epsilon_f$$

With constant C_1. The identification algorithm with the cosine window is indeed robustly convergent and the convergence rate of the worst case error is $O(1/n^2)$.

Triangular window: window for *Cesaro sum* based identification

$$w_n(k) = 1 - \frac{|k|}{n} \qquad (-n \le k \le n).$$

The algorithm is robustly convergent and the order of the identified model is less than N. The convergence rate of the model error can be shown to be $O(1/n)$.

Trapezoidal window function: a possible parameterized window function with a trade-off between the approximation error and the noise error in terms of parameter m.

$$w_n(k) = \begin{cases} 1 + \dfrac{k}{n-m} & (m-n \le k \le 0) \\ 1 & (0 \le k \le 2m) \\ \dfrac{n+m-k}{n-m} & (2m \le k \le n+m) \\ 0 & \textit{elsewhere} \end{cases}$$

See [Gu 92]. This window is *not* even symmetric with respect to k and therefore does not fit directly into the framework of the results derived in the previous section. However the shifted window $w_n(k-m)$ is even symmetric.
The worst case identification error is

$$e^{N_f} \le \left(\frac{2(n+m)}{n-m}\right)\epsilon_f + \frac{2M}{\rho-1}\left(\rho^{-(N_f-1)} + \rho^{-(N_f-n+m-1)} + 2\rho^{-2m}\right)$$

with $m < n < N_f$. If $m \to N_f$ then the trapezoidal window approaches a one-sided rectangular window which yields an exponential convergence rate for the approximation error. On the other hand if $m \to 0$, then it approaches the triangular window whose noise error does not exceed ϵ_f. If m is chosen such that $m/n \approx c$ where $0 < c < 1$. For each n the value of m is taken to be the largest integer no greater than $n \bullet c$. This window function leads to a robustly convergent algorithm for identification and the worst case error converges at the rate of $O(1/n)$.

Other (nonlinear) algorithms for the second stage of the two-stage algorithm can be found in for example [Mäkilä 92], [Akçay 93] and [Jacobson 93].

7.3. A linear algorithm for the second stage

7.3.1 Polynomial approximation

When the frequency response data E_f^{Nf} are interpolated by a polynomial with coefficients p_k, without the use of a window function then also an identified model $P_{Nf}(E_f^{Nf})(z)$ results. This algorithm operates linearly on the frequency response data. However, the induced norm of an ordinary polynomial interpolation is not bounded by 1, as was the case for spline interpolation in lemma 7.2. The error of the polynomial interpolation diverges in the face of corrupted data. The high frequency noise causes a lack of correlation in the frequency response data and this leads to divergent interpolations. Since the data is generated by a physical system the samples should be correlated and so a convergent interpolation is known to exist, but the solution does not depend continuously on the data. This is an example of what is called an *ill posed problem* in [Helmicki 90b]. Often correlation ('continuity') may be restored by bringing a priori information into the algorithm. An observation which can be made is that the divergent sequence of identified models does not lie in the model set defined by the a prior information $\mathcal{H}(D_\rho,M)$. This can be circumvented by constraining the identified model to satisfy the a priori information. A constrained optimization problem has to be solved.
The prior information assumed here makes it natural to constrain the identified model to lie in the Hardy space $\mathcal{H}(D_\rho,M)$. However, in the noise free case, polynomial interpolation is very close to being an orthogonal projection of the original function on a finite dimensional subspace. Therefore a more tractable set to consider is the Hilbert space $\mathcal{H}_2(D_\rho)$. The model set $\mathcal{H}(D_\rho,M)$ is embedded in $\mathcal{H}_2(D_\rho)$ and the 2-norm of the identified model is constrained.

7.3.2 Coefficient approximation

Approximation of the polynomial coefficients subject to a norm constraint.

Let $\varepsilon_f \geq 0$ and $H \in \mathcal{H}(D_\rho,M)$ for $\rho > 1$ and $M < \infty$. Let $\{p_k\}_{k=0}^{Nf-1}$ denote the coefficients of $P_{Nf}(E_f^{Nf})$. Form the function

$$\hat{H}_{id}^{N_f}(z) = \sum_{k=0}^{N_f-1} c_k z^k$$

where the coefficients c_k are defined as the solution of the constrained minimization problem

$$\min_{c_k} \left[\sum_{k=0}^{N_f-1} |c_k - p_k|^2 \right]^{\frac{1}{2}} \quad \text{subject to } \|\hat{H}_{id}^{Nf}\|_2 \leq M.$$

7.3.3 Approximation of functional values

Another way to restore correlation in the ill posed problem of polynomial approximation is to attempt to duplicate the functional evaluation of the interpolating polynomial subject to a

norm constraint.

Let $\varepsilon_f \geq 0$ and $H \in \mathcal{H}(D_\rho, M)$ for $\rho > 1$ and $M < \infty$. Form the function $\hat{H}_{id}^{N_f}(z) = \sum_{k=0}^{N_f-1} f_k z^k$

where the coefficients f_k are defined as the solution of the constrained minimization problem

$$\min_{f_k}\left[\frac{1}{N_f}\sum_{k=1}^{N_f} |\hat{H}_{id}^{N_f}(W_{N_f}^{k-1}) - (E_f^{N_f}(H,w))_k|^2\right]^{\frac{1}{2}} \quad \text{subject to } \|\hat{H}_{id}^{N}\|_2 \leq M.$$

7.3.4 The identification error

The algorithms are linear functions of the frequency response data and depend explicitly on the a prior information ρ, M and ε_f and so the linear algorithms are not robustly convergent. In the absence of noise, both methods asymptotically tend to polynomial interpolation as the number of data increases ($N_f \to \infty$).

The error for coefficient approximation is $\sup_{\substack{H\in\mathcal{H}(D_\rho,M)\\ w\in B_{N_f}(\epsilon_f)}} \|H - \hat{H}_{id}^{N_f}\|_\infty \leq K\tilde{\epsilon}^{\alpha}$

where $\tilde{\epsilon} := \min\left\{\frac{4M\pi}{(\rho-1)N_f}, \frac{M\pi^2(\rho+1)}{(\rho-1)^2 N_f^2}\right\} + \frac{2M}{\rho^{N_f-1}(\rho-1)} + \epsilon_f$

$$K := \left(\frac{\rho+1}{\rho-1}\right)(2M)^{\beta}, \quad \beta := \frac{\log\left(\frac{1}{2}(\rho+1)\right)}{\log(\rho)} \quad \text{and} \quad \alpha := \frac{\log(2\rho)-\log(\rho+1)}{\log(\rho)}.$$

with

The error for approximation of functional values is $\sup_{\substack{H\in\mathcal{H}(D_\rho,M)\\ w\in B_{N_f}(\epsilon_f)}} \|H - \hat{H}_{id}^{N_f}\|_\infty \leq K\tilde{\epsilon}^{\alpha}$

where $\tilde{\epsilon} := \frac{2M}{\rho^{N_f-1}(\rho-1)} + \epsilon_f$ and with the same values for K, β and α as with coefficient

approximation.
See [Helmicki 90b] for the proof of these expressions. In [Akçay 93] another linear algorithm is treated. This is also not robustly convergent, but the worst case error diverges very slowly. It is shown that there does not exist a linear robustly convergent algorithm for this specific problem.

7.4 Implementation in Matlab

The given class of algorithms are well suited to numerical implementation. A start has been made with the implementation of the nonlinear algorithm of Helmicki e.a. of §7.2. The point frequency response identification can be done by using N_f times the algorithm of §7.1.1. The implementation is straightforward. Another possibility is the use of a (smoothed) ETFE model, as suggested in the introduction of this chapter. A procedure to determine ETFE models, which includes smoothing with a window, is available in Matlab.
For choices of N_f commensurate with a power of 2, step 1 of the second stage can be implemented using a sequence of inverse FFT operations. The window functions of §7.2.2 can be used to create different identification algorithms. Although it is doubtful whether it works, the AAK approximation in step 2 with the exact cancellation of the negative terms has been implemented. The FIR approximation to compensate for the noncausal part of the L_∞ approximation and arrive at a H_∞ approximation is more suitable for implementation and is also faster. Also the corresponding error bounds are implemented.
The linear algorithm with polynomial approximation is implemented with both ways, coefficient approximation and approximation of functional values, to overcome the ill posed problem. In Matlab it is possible to solve optimization problems constrained by a norm, but like all optimization problems, convergence to the true optimal value can never be guaranteed. Also the worst case errors of these algorithms are available.
The implementation has not been completed and has not been tested. It is recommendable to complete the implementation of these algorithms and try it out because the algorithms are a promising approach of control oriented system identification.

8. Conclusions and recommendations

As could be expected from approaches which try to find expressions for model errors occurring in identification algorithms, which were not designed to yield this kind of information, the resulting algorithms are not very practical. The discrete implementation of a continuous algorithm, such as the one with the PDF for the Frequency Function, which consists of a lot of (inverse) Fourier transforms and integrals could be expected to be troublesome. For the differentiation and primitivation operations first order methods are used. It is likely that, if high order differentials or primitives have to be computed when the error algorithm is used with high-order models, some bad initial conditions will blow up and lead to very erroneous results. This is for example the case in the regression vector, so when the results of the routine are unexpected, it makes sense to check (plot) this vector for verification. The routines could be improved by more optimal high quality numerical methods, but these algorithms are time-consuming. A very critical part is the construction of a *bounded* covariance function of the undermodelling, which has to be used in IDFT's. This function determines largely the envelope of the resulting error function. The original covariance function is bounded with a window, but through this operation the high frequency information is lost, so the resulting covariance is only an approximation. The remark of §4.3.5 on the caution of the use of filters with ARX models is an interesting notion for everybody who has to deal with this kind of problems. The number of samples which can be used and the order of the models are limited in practice because the large matrices could give memory problems on an ordinary pc.

In the first part of this report it has been shown that prior assumptions on the likely nature of the undermodelling $G_{\Delta,0}$ indeed can be translated into probable influences on the optimal estimate of the parameters $\hat{\Theta}_N$. The undermodelling error, which is a deterministic quantity in classical identification theory, became also a random variable which fits in the stochastic embedding of the identification.

The major problem with the algorithms in a stochastic embedding is that the transfer function of the model should be parametrized linearly in the parameters Θ on which the error algorithm has to operate. This is the case with FIR models and for this kind of model the (implemented) error algorithm works quite good as seen in §5.1. The algorithm has been made suitable for ARX models, but in case of a PDF for the frequency function of the undermodelling the algorithm is a (coarse) approximation. The algorithm with a PDF for the impulse response of the undermodelling ([Goodwin 92]) is said to be applicable to ARX models (and the FIR

models are a specific form of the ARX models), but the algorithm only operates on the parameters of the numerator and so the denominator is considered to be (exactly) good. In the implementation the standard ARX identification procedure is used which generates also errors in the denominator, but these are left out of the error algorithm.
Since a model which is applied to the error algorithm has to be parametrized linearly in the parameters also most other model structures, such as the *Minimal Polynomial Start Sequence of Markov parameters* (MPSSM) model, are not suitable for use with the algorithm.
The availability of Fisher information matrix M_ξ to compute the Cramer-Rao lowerbounds of the model error and the simple form of the likelihood function ℓ in the case of Gaussian embedding is a motivation to use the Gaussian assumption in GOODWIN2. In case of the Gaussian assumption, the maximization problem is in principle convex (see §3.2), but it can never be guaranteed that the true maximum will be found. In practical situations it will generally also be impossible to estimate a true value for the covariance of the noise, so then a value has to be specified by the user.
As a result of all this, it takes a lot of pragmatical thinking to apply the algorithm to an example, especially if this concerns a difficult ball balancing process.

The problem formulation for deterministic, Robust Identification algorithms of Helmicki e.a. which has the goal to identify a model and simultaneously determine the worst case model error, seems to be suitable for control-oriented system identification. The principles of the deterministic robust identification algorithms of chapter 7 offer good prospects. The algorithms for the model and the corresponding error bounds are simple and are suitable for straightforward implementation. The nonlinear algorithms are robustly convergent. The linear algorithms which have been presented do not have this property. A disadvantage is the large order of the resulting models, because in (robust) control, low order (linear) models are used by preference. From this point of view the tag of 'control-oriented system identification algorithms' is doubtful. On the other hand, the simple algebraic expressions for the worse case model errors of the different algorithms seem to be applicable for controller design and the implementation of the expressions is extremely easy.
The advantage of spline interpolation and the corresponding sinusoidal window is that the error bound due to undermodelling and the error bound due to noise are decoupled, so as $N \to \infty$ only the noise bound remains. In general it is not possible to reduce the undermodelling error and noise error simultaneously and there is a trade-off between noise reduction and system approximation. The trapezoidal window function of §7.2.2 offers possibilities to incorporate the trade-off between noise and undermodelling as a design parameter.
The implementation of these algorithms has not been completed, but it should not take a long time to finish it, not considering the (graphical) user interface.

One of the conclusions of the literature research, which has been done on account of the library assignment, is that there is an increasing amount of literature about the subject of

'Robust identification'. At the moment, books on this subject have not yet been signalized, but a lot of articles appear and several theses have been written. The report of the library assignment ([library assignment]) contains literature which has appeared more recently, but which has not been used for this Master's Thesis. This could be a basis for further research on this increasingly important subject.

The general and closing conclusion of this Master's Thesis report is that the problem of finding algorithms for model errors and for Robust Identification is an interesting and instructive subject from the theoretical identification point of view. However with the goal of this Master's Thesis to only analyze and evaluate certain algorithms and implement them in Matlab, the resulting routines for the stochastic embedding are not suitable for addition to for example the Matlab 'Identification toolbox'. It will take further literature research and probably own synthesis of algorithms to get expressions and corresponding routines which are really generally applicable. The 'new' approach of Helmicki e.a. is promising and the algorithms in literature concentrate on this problem.

Literature list

[Akcay 93] Akcay, H. and G. Gu; P.P. Khargonekar
'A class of algorithms for identification in H_∞: continuous-time case'
IEEE Transactions on Automatic Control, USA, Vol: 38, 1993, Iss: 2, p. 289-94

[Goodwin 89] Goodwin, G.C. and M.E. Salgado
'A stochastic embedding approach for quantifying uncertainty in the estimation of restricted complexity models'
International Journal of Adaptive Control and Signal Processing, UK, Vol: 3, 1989, Iss: 4, p. 333-56

[Goodwin 92] Goodwin, G.C. and M. Gevers; B. Ninness
'Quantifying the error in estimated transfer functions with application to model order selection'
IEEE Transactions on Automatic Control, USA, Vol: 37, 1992, Iss: 7, p. 913-28

[Gu 92] Guoxiang Gu and P.P. Khargonekar
'A class of algorithms for identification in H_∞'
Automatica, UK, Vol: 28, 1992, Iss: 2, p. 299-312

[Helmicki 90a] American Control Conference, Proceedings of the 1990 (p. 386-91 vol.1)
Helmicki, A.J. and C.A. Jacobson; C.N. Nett
'Identification in H_∞: a robustly convergent, nonlinear algorithm'
Conf. Loc: San Diego, CA, USA, Conf. Date: 23-25 May 1990, American Autom. Control Council, Green Valley, AZ, USA, Date: 1990

[Helmicki 90b] American Control Conference, Proceedings of the 1990, (p. 2418-23 vol.3)
Helmicki, A.J. and C.A. Jacobson; C.N. Nett
'Identification in H_∞: linear algorithms'
Conf. Loc: San Diego, CA, USA, Conf. Date: 23-25 May 1990, American Autom. Control Council, Green Valley, AZ, USA, Date: 1990

[Helmicki 91] Helmicki, A.J. and C.A. Jacobson; C.N. Nett
'Control oriented system identification: a worst-case/deterministic approach in H_∞'
IEEE Transactions on Automatic Control, USA, Vol: 36, 1991, Iss: 10, p. 1163-76

[Jacobson 93] American Control Conference, Proceedings of the 1993 (p. 1539-43 vol.2)
Jacobson, C.A. and G. Tadmor
'A note on H_∞ system identification with probabilistic a priori information'
Conf. Loc: San Francisco, CA, USA, Conf. Date: 2-4 June 1993, American Autom. Control Council, Evanston, IL, USA, Date: 1993

[Makila 92] Makila, P.M. and J.R. Partington
'Robust identification of strongly stabilizable systems'
IEEE Transactions on Automatic Control, USA, Vol: 37, 1992, Iss: 11, p. 1709-16

Background literature

[Library assignment] van Riel, N.A.W.
'report of the library assignment for Master's Thesis: Algorithms for estimation of model errors and uncertainties arising in black box identification'
Eindhoven University of Technology, 1995

[Ljung 87] Ljung, L.
'System identification: theory for the user'
Englewood Cliffs: Prentice Hall, 1987

[Stochastische Systeemtheorie] van de Bosch, P.P.J. and A.C. van der Klauw
'Stochastische systeemtheorie: modeling, Identification and simulation of dynamical systems'
College diktaat, Eindhoven University of Technology, 1994

[Toegepaste Systeemanalyse] Backx, A.C.P.M. and A.J.W. van den Boom
'Toegepaste Systeemanalyse'
College diktaat, Eindhoven University of Technology, 1994

About Matlab

[Ident] Ljung, L.
'System identification toolbox - user's guide',1991

[Matlab] 'Matlab: high-performance numeric computation and visualization software'
Reference guide, The Math Works Inc., 1992

Appendix A: Most relevant norms and normed spaces

Consider a signal x(k).

- Amplitude / ∞ norm:

$$\|x\|_\infty = \sup_{k \in Z} |x(k)|$$

- Energy / 2 norm:

$$\|x\|_2 = \sqrt{\sum_{k=-\infty}^{\infty} |x(k)|^2}$$

- 1 norm:

$$\|x\|_1 = \sum_{k=-\infty}^{\infty} |x(k)|$$

The space of the discrete finite amplitude signals:
$\ell_\infty = \{x \mid \|x\|_\infty < \infty\}$ (continuous time: $L_\infty = \{x \mid \|x\|_\infty < \infty\}$)
The space of the discrete finite energy signals:
$\ell_2 = \{x \mid \|x\|_2 < \infty\}$ (continuous time: $L_2 = \{x \mid \|x\|_2 < \infty\}$)

The H_∞ norm for SISO systems

Consider unit energy inputs $u \in L_2$, $\|u\|_2 \leq 1$ and maximize the energy $\|y\|_2$ of output y.

$$\|H\|_\infty = \sup_{\substack{u \in L_2 \\ \|u\|_2 \leq 1}} \|y\|_2 = \sup_{U \in L_2} \frac{\|HU\|_2}{\|U\|_2} = \max_\omega |H(j\omega)|$$

So this is the peak gain in, for example, a Bode diagram.

Appendix B: The Discrete Fourier Transform

The Discrete Fourier Transform (DFT) and it's numerical implementation Fast Fourier Transform (FFT) compute the sampled spectrum $X(\omega)$ of a signal x(k). The resulting DFT is a periodical function with period 2π in discrete frequency domain (sidebands are introduced). Sampling in the frequency space and then performing the backward operation (IDFT) leads to a periodical signal $\tilde{x}(k)$ in the time space (with period N). So sampling in the frequency domain is only allowed if the signal is bounded in time and frequency. Then no aliasing occurs and one period of $\tilde{x}(k)$ is equal to the original signal x(k). ($\tilde{x}(k)$ Is the summation of replications of x(k) which are translated over N.)
In the transformation from the continuous s-plane to the discrete one, the s-plane may be divided into a *primary strip*, which corresponds to the central band, between $-0.5j\omega_s$ and $+0.5j\omega_s$, and *complementary strips* which correspond to the sidebands.

The Z-plane

Usually the Z-transformation is defined with negative powers of z:

$$F(z) = Z\{f(k)\} = \sum_{k=0}^{\infty} f_k z^{-k}$$

but in the articles and in this report, the Z-transform is defined with positive powers of z:

$$F(z) = \sum_{k=0}^{\infty} f_k z^{k}$$

The relation between the continuous frequency s-domain and the discrete frequency z-domain:

$z = e^{st_s}$ where t_s is the sample time. So poles or zeros $s = -a \pm jb$ map to the position $z = e^{(-a\pm jb)ts} = e^{-ats}e^{\pm jbts}$. The magnitude (i.e. the distance to the origin) is e^{-ats}. The angles with the positive real axis of the z-plane, measured positive in the counterclockwise direction, are $\pm bt$ radians.
1. Stability: In the s-plane is the imaginary axis (a=0) the boundary of the stable pole region (for stability $a > 0$). This axis maps into $z = e^{\pm jbts}$, a unit circle about the origin. So $a > 0$ for stability means that all poles must lie **out**side the unit circle in the z-plane.
2. Time constant τ: $\tau = 1/|a|$, so in the z-plane the poles must lie on a circle of radius e^{-ats}. Moving the poles further from the origin of the z-plane increases the speed of response.
3. Sampling frequency: as the imaginary part b of the poles moves closer to the limit $\omega_s/2$ of the primary strip, the number of samples per cycle reduces to the minimum of 2. In the z-plane, this reduction occurs as the angle $e^{\pm jbts}$ of the poles moves closer to the direction of the negative real axis.

There are several approximations to transform transfer function in s-domain to z-domain and

backwards. The most common is the *bilinear transformation* or *Tustin's method*:

$$s = \frac{2}{t_s}\left(\frac{z-1}{z+1}\right).$$

This is based on the trapezoidal rule for approximation of integrators.

Appendix C: Experiment design and data conditions

The most important results of experiment design and data processing which are treated thoroughly in [Stochastische Systeemtheorie] and [Toegepaste Systeemanalyse] are summarised here. There are design variables that have to be chosen a priori to an experiment and there are variables that can be chosen after the data has been collected.
The selection of input(s) and output(s) depends on the kind of model which has to be estimated and of the possibility to excite certain inputs and measure certain outputs. In the situations as described in this report, this is no problem since only SISO systems are considered.

Design of the input signal
The estimated model will be closer to the true system in frequency regions where the input spectrum is large compared to the noise spectrum. The input signal should contain enough 'information'. In this respect ZMWN is the ideal input signal, because it contains all frequencies and the energy is divided equally over all frequencies. However, in practice it is usually not allowed to excite a system with ZMWN. Another good signal for system identification is a Pseudo Random Binary Sequence (PRBS), in which the energy distribution over the frequencies can be influenced.

Choice of the sampling frequency
Two different sample frequencies can be distinguished: the sampling frequency for data collection $\omega_{s,m}$ and the sample frequency that is used in the identification procedure $\omega_{s,i}$. For measurement the sample rate should be as large as possible to collect as much information as possible. A lower bound is of course Shannon's theorem, but the maximal frequency in a process is unknown. A suitable choice for the sampling frequency is: $\omega_{s,m} \geq 10\omega_B$, where ω_B is the *bandwidth of the process*. The bandwidth ω_B of a process is defined as the maximum frequency for which the magnitude of the frequency function reaches the level of $1/\sqrt{2}$ times it's static value (-3dB). To avoid aliasing effects, an anti-aliasing (Low Pass) filter can be added to the system. This can only be a physical, analog (continuous) filter. The smallest time constant of the system determines the sample rate.
A too high sample frequency for identification can lead to numerical problems, since the poles of a discrete system are pushed towards z=1 in the complex z-plane. Another consideration is the use of one-step ahead prediction. Smaller prediction steps will emphasize the high-frequency fit of the model. A practical choice for the sample frequency is $10\omega_B \leq \omega_{s,i} \leq 30\omega_B$.
The sample frequency for data acquisition should be higher than the sample frequency for identification.
The sample frequency of the measured signals has to reduced to get samples which can be used in identification. This is called *decimation*. Before decimation, the signals have to be filtered again by a (digital) anti-aliasing filter.

Length of the experiment

The duration of an experiment T should be chosen such that the largest time constant fits several times in this time. The combination of the sample frequency and the duration determines the number of samples N.

Data (pre)processing

Data (pre)processing concerns:

- removal of outliers, spikes, offsets and trends;
- scaling of the signals to equalize their variance.

Choice of model structure and order and eventual choice of the prefilter

The choice of the model structure and order depends on the intended model application. In the Matlab 'System Identification toolbox' functions are available which automatically select the optimal model order (smallest Cost Function).

The prefilter F(s) should be designed such that the frequency region of interest is emphasized. The *dead time* of a system can be estimated using correlation analysis (see [Stochastische Systeemtheorie] and [Toegepaste Systeemanalyse]). If this prior information is incorporated in identification then the model will be better. If the dead time is underestimated then the first parameters of the B-polynomial will be almost 0. Overestimation of the dead time is worse and results in biased parameters because these parameters have to compensate the incorrectness.